BDeBooks.com

E-BOOK

www.BDeBooks.com
FB.com/BDeBooksCom
BDeBooks.Com@gmail.com

বাংলাদেশ কারিগরি শিক্ষা বোর্ড কর্তৃক প্রবর্তিত ৪ বছর মেয়াদি প্রকৌশল ডিপ্লোমা শিক্ষাক্রমের চতুর্থ পর্বের কম্পিউটার, কম্পিউটার সাইন্স অ্যান্ড টেকনোলজি, ডাটা টেলিকমিউনিকেশন অ্যান্ড নেটওয়ার্কিং টেকনোলজির নতুন পাঠ্যক্রম অনুযায়ী প্রণীত

প্রোগ্রামিং ল্যাংগুয়েজ-২

Programming Language-2

Subject Code : 6642

রচনায়

প্রকৌশলী মোহাম্মদ মাহবুব আলম

বিএসসি ইন কম্পিউটার সাইন্স অ্যান্ড ইঞ্জিনিয়ারিং

ইনস্ট্রাক্টর (কম্পিউটার)

কুমিল্লা পলিটেকনিক ইনস্টিটিউট, কোটবাড়ী, কুমিল্লা

প্রাক্তন জুনিয়র ইনস্ট্রাক্টর (কম্পিউটার)

ফেনী পলিটেকনিক ইনস্টিটিউট, ফেনী

পলিটেকনিক ইনস্টিটিউট-এর ছাত্রছাত্রীদের জন্য প্রণীত বইসমূহ ঃ

ডাটাবেস ম্যানেজমেন্ট, প্রোগ্রামিং-ইন-C; প্রোগ্রামিং-ইন-C++

কম্পিউটার পেরিফেরালস্; ডাটা কমিউনিকেশন অ্যান্ড কম্পিউটার নেটওয়ার্ক-১

কম্পিউটার অপারেটিং সিস্টেম; ভিজ্যুয়াল প্রোগ্রামিং

উচ্চ মাধ্যমিক ভোকেশনাল ছাত্রছাত্রীদের জন্য প্রণীত বই ঃ

কম্পিউটার অপারেশন অ্যান্ড মেইনটেন্যান্স-২

সম্পাদনায়

ইঞ্জিনিয়ার মোঃ ইমদাদুল হক

অধ্যক্ষ

নোয়াখালী আইডিয়াল পলিটেকনিক ইনস্টিটিউট, নোয়াখালী

হক পাবলিকেশনস্

HAQUE PUBLICATIONS

৩৮ বাংলাবাজার (২য় তলা), ঢাকা-১১০০

প্রকাশক
হক পাবলিকেশনস্-এর পক্ষে
মিসেস জাহানারা হক
৩৮ বাংলাবাজার (২য় তলা)
ঢাকা-১১০০
ফোন ঃ ৯৫৮০৩৭০

প্রথম প্রকাশ ঃ ২ মে, ২০১২
দ্বিতীয় প্রকাশ ঃ ১ আগস্ট, ২০১৪

প্রচ্ছদ পরিকল্পনায় ঃ ডিজাইন প্লাস

বর্ণবিন্যাসে
জি. মাওলা কম্পিউটারস্
৩৮ বাংলাবাজার (৫ম তলা)
ঢাকা-১১০০

মুদ্রণে
আরাম প্রিন্টিং প্রেস
কাগজীটোলা, সূত্রাপুর
ঢাকা-১১০০

ISBN : 978-984-8980-06-4

মূল্য (MRP) ঃ ২০০.০০ টাকা মাত্র

ভূমিকা

বর্তমান বিশ্বের সর্বাধুনিক প্রযুক্তি হচ্ছে কম্পিউটার প্রযুক্তি। কারণ শিল্পকারখানা, গবেষণা, শিক্ষা প্রতিষ্ঠান, চিকিৎসাশাস্ত্র ইত্যাদি সকল ক্ষেত্রই আজ কম্পিউটার অটোমেটেড। কম্পিউটার অটোমেশনের সঙ্গে কম্পিউটার প্রোগ্রামিং ও প্রোগ্রামিং লেখার জন্য প্রোগ্রামিং ল্যাংগুয়েজ অত্যন্ত গুরুত্বপূর্ণ। তাই শিক্ষা ও গবেষণার জন্য প্রোগ্রামিং ল্যাংগুয়েজ অধ্যয়ন একটি অতি আবশ্যকীয় বিষয়। বিষয়টি অত্যন্ত গুরুত্বসহ বিবেচনা করে বর্তমানে কারিগরি শিক্ষার সকল টেকনোলজিতে কম্পিউটার প্রোগ্রামিং ল্যাংগুয়েজ বিষয়কে আবশ্যিক করা হয়েছে। বাংলা মাধ্যমে কম্পিউটার প্রোগ্রামিং ল্যাংগুয়েজ বিষয়ে বইয়ের চাহিদা ও অভাব মেটানোর উদ্দেশ্যে **"প্রোগ্রামিং ল্যাংগুয়েজ-২"** বইটি লেখার প্রয়াস।

আমার এ প্রচেষ্টাকে বাস্তবরূপ দিতে ফেনী পলিটেকনিক ইনস্টিটিউটের ইলেকট্রিক্যাল বিভাগের প্রধান ও চীফ ইনস্ট্রাক্টর ***জনাব মোঃ নুরুল হক স্যার***-এর অনুপ্রেরণা ও সহযোগিতা চিরস্মরণযোগ্য। বইটি লেখার সময় বইপত্র ও মানসিক অনুপ্রেরণা যোগানোর জন্য আমি তাঁর প্রতি কৃতজ্ঞ। দ্রুত মুদ্রণের ব্যবস্থা করায় **"হক পাবলিকেশনস্"**-এর স্বত্বাধিকারিণী মিসেস জাহানারা হককে ধন্যবাদ জানাচ্ছি।

তড়িৎ গতিতে বইটি সম্পাদনা করতে গিয়ে যে কোন ত্রুটির ব্যাপারে এবং এর মান সার্বিক বৃদ্ধিকল্পে পাঠক-পাঠিকা ও কম্পিউটার বিশেষজ্ঞদের মতামত ও পরামর্শ এবং যে কোন পরিবর্তন, পরিবর্ধন ও সংশোধন সাদরে গ্রহণ করবো এবং পরবর্তীতে সংশোধিত রূপে উপস্থাপন করার চেষ্টা করবো।

পরিশেষে যাদের উদ্দেশ্যে এ বই প্রকাশিত হল তাদের সামান্যতম উপকারে আসলেই আমার প্রচেষ্টা সার্থক বলে মনে করবো।

ধন্যবাদান্তে

লেখক

প্রকৌশলী মোহাম্মদ মাহবুব আলম

সূচি

অধ্যায়-১ ঃ সি ভাষায় গ্রাফিক্স

১.০	ভূমিকা	১৩
১.০.১	C প্রোগ্রামিং ল্যাংগুয়েজে গ্রাফিক্স প্রোগ্রামিং এর ক্ষেত্রে করণীয়	১৩
১.১	পিক্সেল ও রেজোলুশন	১৪
১.২	উদাহরণসহ বিভিন্ন প্রকার গ্রাফিক্স ফাংশনের ফরম্যাট ও ব্যবহার	১৫
১.৩	ফরম্যাট, ব্যবহার ও উদাহরণসহ বিভিন্ন ধরনের ফাংশন	১৬
১.৪	ইনপুট আউটপুট অপারেশনের ক্ষেত্রে cprintf ( ) ও cscanf ( ) ফাংশনের ব্যবহার	২২
১.৫	গ্রাফিক্স ফাংশন ব্যবহার করে কালার ইমেজ তৈরির প্রোগ্রাম	২৩
১.৬	সি প্রোগ্রামিং ল্যাংগুয়েজে ইমেজ সংরক্ষণ ও ইমেজ লোডিং পদ্ধতি	২৭
১.৭	টেক্সট স্ট্রিংকে স্থান পরিবর্তন করার প্রক্রিয়া	২৯
✪	**অনুশীলনী-১**	
✪	অতি সংক্ষিপ্ত প্রশ্নোত্তর	৩০
✪	সংক্ষিপ্ত প্রশ্নাবলি	৩২
✪	রচনামূলক প্রশ্নাবলি	৩৩

অধ্যায়-২ ঃ সি ভাষায় প্রোগ্রাম ইউটিলিটিজ

২.০	ভূমিকা	৩৪
২.০.১	প্রোগ্রাম ইউটিলিটিজসমূহ	৩৫
২.১	সফটওয়্যার ইন্টারাপ্ট পদ্ধতি ব্যবহার করে ফাংশন কী ডিটেক্ট করার মূলনীতি	৩৬
২.২	সি ভাষায় সিস্টেম টাইমারের ব্যবহার	৩৮
২.৩	সি ভাষায় মাউস পয়েন্টার তৈরির পদ্ধতি	৩৯
২.৪	গ্রাফিক্স মোডে টেক্সট ও নাম্বার প্রদর্শন পদ্ধতি	৪১
২.৫	সি ভাষায় অবজেক্ট স্থানান্তর পদ্ধতি	৪২
২.৬	সি ভাষায় ত্রিমাত্রিক অবজেক্ট তৈরির পদ্ধতি	৪৪
✪	**অনুশীলনী-২**	
✪	অতি সংক্ষিপ্ত প্রশ্নোত্তর	৪৮
✪	সংক্ষিপ্ত প্রশ্নাবলি	৪৯
✪	রচনামূলক প্রশ্নাবলি	৪৯

অধ্যায়-৩ ঃ অবজেক্ট অরিয়েন্টড প্রোগ্রামিং

৩.০	ভূমিকা	৫০
৩.১	সফটওয়্যার বিবর্তন	৫১
৩.১.১	প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং ও অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর মধ্যে পার্থক্য	৫৪
৩.২	ট্র্যাডিশনাল প্রোগ্রামিং এর অসুবিধা	৫৫
৩.৩	অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর বিভিন্ন টার্মসমূহ	৫৬
৩.৪	অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজের তালিকা	৬১
৩.৫	অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর সুবিধা-অসুবিধা	৬২
৩.৬	অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর ব্যবহার	৬৩
✪	**অনুশীলনী-৩**	
✪	অতি সংক্ষিপ্ত প্রশ্নোত্তর	৬৪
✪	সংক্ষিপ্ত প্রশ্নাবলি	৬৬
✪	রচনামূলক প্রশ্নাবলি	৬৭

অধ্যায়-৪ ঃ সি++ প্রোগ্রামিং

৪.০	ভূমিকা ও ইতিহাস	৬৮
৪.১	অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং হিসাবে সি++ এর বৈশিষ্ট্যসমূহ	৬৯
৪.২	সি++ প্রোগ্রাম স্ট্রাকচার	৬৯
৪.৩	সি++ ইনপুট-আউটপুট স্ট্রিম ও সি++ ক্লাস স্ট্রিম	৭১
৪.৩.১	সি++ ইনপুট-আউটপুট ফরম্যাট	৭৪
৪.৪	সি++ এ ব্যবহৃত কীওয়ার্ড, ভেরিয়েবল, কন্সট্যান্ট, অপারেটর ও কন্ট্রোল স্টেটমেন্টসমূহ.	৮২
৪.৪.১	সি++ ফাংশন	৮৯
৪.৫	সি++ ল্যাংগুয়েজে ক্লাস	৯০
৪.৬	সি++ ভাষায় ফাংশন ওভারলোডিং	৯৫
৪.৬.১	অপারেটর ওভারলোডিং	৯৭
৪.৭	সি++ ভাষায় কনস্ট্রাক্টর, কপি কনস্ট্রাক্টর, ডেস্ট্রাক্টর ও ফ্রেন্ড ফাংশন	৯৯
৪.৮	সি++ ভাষায় ইনহেরিটেন্স	১০৬
✪	**অনুশীলনী-৪**	
✪	অতি সংক্ষিপ্ত প্রশ্নোত্তর	১১৫
✪	সংক্ষিপ্ত প্রশ্নাবলি	১২০
✪	রচনামূলক প্রশ্নাবলি	১২১

অধ্যায়-৫ ঃ জাভা ও জাভার বৈশিষ্ট্য

৫.০	ভূমিকা	১২৩
৫.১	জাভা ভাষার ইতিহাস	১২৩
৫.১.১	জাভার বৈশিষ্ট্য	১২৪
৫.২	জাভা ডেভেলপমেন্ট এনভায়রনমেন্ট এর ধাপসমূহ	১২৭
৫.২.১	জাভা ভার্চুয়াল মেশিন	১২৯
৫.৩	জাভা ও সি++ ল্যাংগুয়েজের মধ্যে পার্থক্য	১৩০
৫.৪	জাভার প্রয়োগক্ষেত্র বা ব্যবহার	১৩১
৫.৫	জাভা ভাষার প্রোগ্রামিং স্টাইল ও প্রোগ্রাম লেখার নিয়মাবলি	১৩৩
৫.৬	জাভায় ব্যবহৃত ফাঁকা স্থান, আইডেন্টিফায়ার, লিটারেলস, কমেন্টস, সেপারেটর ও কীওয়ার্ড	১৩৪
৫.৭	জাভা প্রোগ্রামের স্ট্রাকচার	১৩৮
✪	**অনুশীলনী-৫**	
✪	অতি সংক্ষিপ্ত প্রশ্নোত্তর	১৪১
✪	সংক্ষিপ্ত প্রশ্নাবলি	১৪২
✪	রচনামূলক প্রশ্নাবলি	১৪৩

অধ্যায়-৬ ঃ ডাটা টাইপ, ভেরিয়েবল, অপারেটর কন্ট্রোল স্টেটমেন্ট ও অ্যারে

৬.০	ভূমিকা	১৪৪
৬.১	জাভা ভাষায় ডাটা টাইপ	১৪৪
৬.২	ভেরিয়েবল ডিক্লারেশন ও ডাইনামিক ইনিশিয়ালাইজেশন	১৪৯
৬.৩	ভেরিয়েবলের স্কোপ ও লাইফটাইম	১৫১
৬.৪	জাভায় ব্যবহৃত অপারেটরসমূহ	১৫১
৬.৫	জাভা ভাষার কন্ট্রোল স্টেটমেন্টসমূহ	১৬১
৬.৬	জাভা ভাষায় অ্যারে	১৭৬
✪	**অনুশীলনী-৬**	
✪	অতি সংক্ষিপ্ত প্রশ্নোত্তর	১৮০
✪	সংক্ষিপ্ত প্রশ্নাবলি	১৮২
✪	রচনামূলক প্রশ্নাবলি	১৮৩

অধ্যায়-৭ ঃ ক্লাস, অবজেক্ট, মেথড ও কন্সটাক্টর

৭.০	ভূমিকা	১৮৪
৭.১	জাভা ভাষায় ক্লাস ও অবজেক্ট ডিক্লারেশন	১৮৪
৭.১.১	Class মেম্বার এক্সেসিং	১৮৬
৭.২	অবজেক্ট রেফারেন্স ভেরিয়েবল এসাইন করার প্রক্রিয়া	১৮৯
৭.৩	সিনটেক্সসহ মেথড	১৮৯
৭.৪	ক্লাসে মেথড সংযোজনের প্রক্রিয়া	১৯১
৭.৫	মেথড ব্যবহারের সুবিধাসমূহ	১৯৪
৭.৬	ওভারলোডিং মেথড	১৯৪
৭.৭	কনস্ট্রাক্টর ও ওভারলোডিং কনস্ট্রাক্টর	১৯৫
৭.৮	ইন্সটেন্স ভেরিয়েবল হাইডিং ও গারবেজ কালেকশন	১৯৯
৭.৯	ক্লাস, অবজেক্ট, মেথড ও কনস্ট্রাক্টর ব্যবহার করে প্রোগ্রাম	১৯৯
✪	**অনুশীলনী-৭**	
✪	অতি সংক্ষিপ্ত প্রশ্নোত্তর	২০০
✪	সংক্ষিপ্ত প্রশ্নাবলি	২০১
✪	রচনামূলক প্রশ্নাবলি	২০২

অধ্যায়-৮ ঃ ইনহেরিটেন্স পলিমরফিজম

৮.০	ভূমিকা	২০৩
৮.১	সুপার ক্লাস ও সাব ক্লাস	২০৩
৮.২	বিভিন্ন প্রকার ইনহেরিটেন্স	২০৪
৮.৩	জাভা ভাষায় ওভাররিডেন মেথড	২০৯
৮.৪	জাভায় ডাইনামিক রান টাইম পলিমরফিজম	২১১
৮.৫	জাভায় অ্যাবস্ট্রাক্ট ক্লাস	২১২
৮.৬	ফাইনাল কীওয়ার্ড	২১৩
৮.৭	ইনহেরিটেন্স ও পলিমরফিজম ব্যবহার করে প্রোগ্রাম	২১৩
✪	**অনুশীলনী-৮**	
✪	অতি সংক্ষিপ্ত প্রশ্নোত্তর	২১৪
✪	সংক্ষিপ্ত প্রশ্নাবলি	২১৫
✪	রচনামূলক প্রশ্নাবলি	২১৫

অধ্যায়-৯ ঃ প্যাকেজ ও ইন্টারফেস

৯.০	ভূমিকা	২১৬
৯.১	সিনটেক্সসহ প্যাকেজের সংজ্ঞা	২১৬
৯.২	প্যাকেজ ডিক্লারেশনের প্রয়োজনীয়তা	২১৮
৯.৩	বিভিন্ন ধরনের প্যাকেজের ফাংশন	২১৯
৯.৪	প্যাকেজের স্ট্রাকচার	২২০
৯.৫	সিনটেক্সসহ ইন্টারফেস	২২০
৯.৬	ইন্টারফেস বাস্তবায়ন প্রক্রিয়া	২২১
৯.৭	মাল্টিপল ইন্টারফেস	২২২
৯.৮	প্যাকেজ ও ইন্টারফেস ব্যবহার করে প্রোগ্রাম	২২৩
✪	**অনুশীলনী-৯**	
✪	অতি সংক্ষিপ্ত প্রশ্নোত্তর	২২৪
✪	সংক্ষিপ্ত প্রশ্নাবলি	২২৫
✪	রচনামূলক প্রশ্নাবলি	২২৫

অধ্যায়-১০ ঃ মাল্টিথ্রেড প্রোগ্রামিং

১০.০	ভূমিকা	২২৬
১০.১	সিনটেক্সসহ মাল্টিথ্রেড প্রোগ্রামিং	২২৬
১০.২	প্রসেস বেইজড মাল্টিটাস্কিং ও থ্রেড বেইজড মাল্টিটাস্কিং এর মধ্যে পার্থক্য	২২৭
১০.৩	স্টেট ডায়াগ্রামসহ থ্রেড ক্লাসের বিভিন্ন মেথডসমূহ	২২৮
১০.৪	থ্রেড তৈরির পদ্ধতি	২৩০
১০.৫	থ্রেডসমূহের অগ্রগণ্যতা	২৩৩
১০.৬	থ্রেড সিনক্রোনাইজেশন ও ইন্টারথ্রেড কমিউনিকেশন	২৩৫
১০.৭	থ্রেড সাসপেন্ডিং, রিসিউমিং ও স্টপিং	২৩৮
১০.৮	মাল্টিথ্রেড প্রোগ্রামিং মেথড ব্যবহার করা প্রোগ্রাম	২৩৮
✪	**অনুশীলনী-১০**	
✪	অতি সংক্ষিপ্ত প্রশ্নোত্তর	২৪০
✪	সংক্ষিপ্ত প্রশ্নাবলি	২৪২
✪	রচনামূলক প্রশ্নাবলি	২৪২

অধ্যায়-১১ ঃ এনিউমিরেশন, অটোবক্সিং ও অ্যানোটেশনস

১১.০ ভূমিকা ২৪৩
১১.১ সিনটেক্সসহ এনিউমিরেশন ২৪৩
১১.২ ভ্যালু ও ভ্যালু অব মেথডের মধ্যে পার্থক্য ২৪৪
১১.৩ জাভা বাসায় টাইপ র‍্যাপার ২৪৪
১১.৪ বিভিন্ন ধরনের অটোবক্সিং/আন বক্সিং ২৪৫
১১.৫ অ্যানোটেশনস ও এর উদাহরণ ২৪৬
১১.৬ জাভায় ব্যবহৃত বিল্ট ইন অ্যানোটেশনস ২৪৭
১১.৭ অ্যানোটেশন ডিক্লারেশনে সতর্কতাসমূহ ২৪৮
✪ **অনুশীলনী-১১**
✪ অতি সংক্ষিপ্ত প্রশ্নোত্তর ২৪৯
✪ সংক্ষিপ্ত প্রশ্নাবলি ২৫০
✪ রচনামূলক প্রশ্নাবলি ২৫০

অধ্যায়-১২ ঃ ব্যবহারিক

✪ **সি প্রোগ্রামসমূহ**
১। আয়তক্ষেত্র অংকণের প্রোগ্রাম ২৫১
২। বৃত্ত অংকণের সি প্রোগ্রাম ২৫১
৩। উপবৃত্ত (ellipse) অংকণের প্রোগ্রাম ২৫২
৪। বার (bar) অংকণের সি প্রোগ্রাম ২৫২
৫। 3D বার অংকণের সি প্রোগ্রাম ২৫২
৬। বহুভুজ অংকণের প্রোগ্রাম ২৫৩
৭। ত্রিমাত্রিক অবজেক্ট (ডিম) অংকণের প্রোগ্রাম ২৫৩
৮। গ্রাফিক্স মোডে টেক্সট (outtext) প্রদর্শনের প্রোগ্রাম ২৫৩
৯। টেক্সট স্ট্রিংকে পর্দায় স্থান পরিবর্তনের (movetext) প্রোগ্রাম ২৫৪
১০। কারেন্ট টাইম (System time) জানার প্রোগ্রাম ২৫৪
১১। কী ডিটেক্ট (Key detetect) করার প্রোগ্রাম ২৫৫
১২। মাউস পয়েন্টার তৈরি ও পয়েন্টার তৈরির অবস্থান জানার প্রোগ্রাম ২৫৬

১৩। মুভিং অবজেক্ট (আয়তক্ষেত্র) তৈরির প্রোগ্রাম..... ২৫৮
১৪। বাংলাদেশের জাতীয় পতাকা অংকণের প্রোগ্রাম..... ২৫৯
১৫। একটি অবজেক্ট (আয়তক্ষেত্র) অংকণ করে উহাকে অ্যারো কী দ্বারা মুভ করার প্রোগ্রাম ... ২৫৯
১৬। ত্রিমাত্রিক অবজেক্ট (3D Object) তৈরির সি প্রোগ্রাম ২৬১
১৭। একটি মুভিং আয়তক্ষেত্রের মধ্যে "Programming Language-2" বাক্যটি প্রদর্শন করবে এমন একটি স্ক্রীন সেভার তৈরির প্রোগ্রাম..... ২৬২
১৮। একটি আয়তাকার ক্ষেত্রের মধ্যে একটি মুভিং বল (moving ball) প্রদর্শন করার সি প্রোগ্রাম..... ২৬৩

✪ **সি++ প্রোগ্রামসমূহ**

১। আয়তক্ষেত্রের ক্ষেত্রফল নির্ণয়ের সি++ প্রোগ্রাম..... ২৬৫
২। বর্গক্ষেত্রের ক্ষেত্রফল নির্ণয়ের সি++ প্রোগ্রাম ২৬৫
৩। বৃত্তের ক্ষেত্রফল নির্ণয়ের সি++ প্রোগ্রাম ২৬৬
৪। লিপইয়ার (Leap Year) নির্ণয়ের সি++ প্রোগ্রাম..... ২৬৬
৫। তিনটি সংখ্যার মধ্যে বৃহত্তম সংখ্যাটি নির্ণয়ের সি++ প্রোগ্রাম ২৬৭
৬। শর্তসাপেক্ষে অসমবাহু ত্রিভুজক্ষেত্রের ক্ষেত্রফল নির্ণয়ের সি++ প্রোগ্রাম..... ২৬৭
৭। দ্বিঘাত সমীকরণের বাস্তব ও সমান, বাস্তব ও অসমান এবং অবাস্তব মূল নির্ণয়ের সি++ প্রোগ্রাম..... ২৬৮
৮। কোন সংখ্যার ফ্যাক্টরিয়াল মান নির্ণয়ের সি++ প্রোগ্রাম..... ২৬৯
৯। ফিবোনাসসি (fibonacci) সিরিজের মান নির্ণয়ের সি++ প্রোগ্রাম..... ২৬৯
১০। কীবোর্ড কোন সংখ্যা ইনপুট নিয়ে সেটি মৌলিক (prime) কিনা তা নির্ণয়ের সি++ প্রোগ্রাম..... ২৭০
১১। n পর্যন্ত সংখ্যাগুলির মধ্যে মৌলিক সংখ্যাগুলি (Prime number) নির্ণয়ের সি++ প্রোগ্রাম ২৭০
১২। 1+2+3+.....+n=? সিরিজের যোগফল নির্ণয়ের সি++ প্রোগ্রাম..... ২৭১
১৩। এক সেট নাম্বার থেকে জোড় সংখ্যাগুলি নির্ণয়ের সি++ প্রোগ্রাম ২৭১
১৪। কোন সংখ্যা ইনপুট নিয়ে উহাকে রিভার্স আকারে দেখানোর সি++ প্রোগ্রাম..... ২৭২
১৫। কীবোর্ড থেকে কোন সংখ্যা ইনপুট নিয়ে উহার ডিজিটগুলোর যোগফল নির্ণয়ের সি++ প্রোগ্রাম..... ২৭২
১৬। কীবোর্ড থেকে দুটি সংখ্যা ইনপুট নিয়ে অদল-বদল (Swapping) করার সি++ প্রোগ্রাম.. ২৭৩

১৭।	ক্লাস ব্যবহার করে n সংখ্যক ডাটাকে Descending order এ সাজানোর সি++ প্রোগ্রাম	২৭৩
১৮।	switch ব্যবহার করে যোগ, বিয়োগ, গুন ও ভাগ করার সি++ প্রোগ্রাম	২৭৫
১৯।	ফাংশন ব্যবহার করে কোন সংখ্যার ফ্যাক্টরিয়াল মান নির্ণয়ের সি++ প্রোগ্রাম	২৭৬
২০।	ক্লাস ব্যবহার করে আয়তক্ষেত্রের ক্ষেত্রফল নির্ণয়ের সি++ প্রোগ্রাম	২৭৭
২১।	ক্লাস ব্যবহার করে শর্তসাপেক্ষে ত্রিভুজক্ষেত্রের ক্ষেত্রফল নির্ণয়ের সি++ প্রোগ্রাম	২৭৭
২২।	ফাংশন ওভারলোডিং (Function Overloading) বা পলিমরফিজম ব্যবহার করে সি++ ..	২৭৮
২৩।	অপারেটর ওভারলোডিং (Operator Overloading) ব্যবহার করে সি++ প্রোগ্রাম	২৭৯
২৪।	ফ্রেন্ড (friend) ফাংশন ব্যবহার করে সি++ প্রোগ্রাম	২৮০
২৫।	কনস্ট্রাক্টর ব্যবহার করে বর্গক্ষেত্রের ক্ষেত্রফল নির্ণয়ের সি++ প্রোগ্রাম	২৮১
২৬।	কপি কনস্ট্রাক্টর (Copy Constructor) ব্যবহার করে সি++ প্রোগ্রাম	২৮২
২৭।	ইনহেরিটেন্স বা সিঙ্গেল ইনহেরিটেন্স (Inheritance/Single Inheritance) ব্যবহার করে একটি সি++ প্রোগ্রাম	২৮৩
২৮।	মাল্টিলেভেল ইনহেরিটেন্স (multilevel inheritance) ব্যবহার করে সি++ প্রোগ্রাম	২৮৪
২৯।	মাল্টিপল ইনহেরিটেন্স (multiple inheritance) ব্যবহার করে সি++ প্রোগ্রাম	২৮৫
৩০।	হাইব্রিড ইনহেরিটেন্স (Hybrid Inheritance) ব্যবহার করে সি++ প্রোগ্রাম	২৮৬
✪	**জাভা প্রোগ্রামসমূহ**	
১।	লিপইয়ার (LeapYear) নির্ণয়ের জাভা প্রোগ্রাম	২৮৮
২।	1 + 2 + 3 + ... + n = ? সিরিজের যোগফল নির্ণয়ের জাভা প্রোগ্রাম	২৮৯
৩।	প্রথম ১০টি সংখ্যার ফিবোনাসসি সিরিজ নির্ণয়ের জাভা প্রোগ্রাম	২৮৯
৪।	২৫ পর্যন্ত মৌলিক সংখ্যাগুলো নির্ণয়ের জাভা প্রোগ্রাম	২৯০
৫।	নিম্নের আউটপুটটি প্রদর্শনের জাভা প্রোগ্রাম ঃ 12345, 1234, 12, 1	২৯০
৬।	কোন সংখ্যাকে ইনপুট নিয়ে উহাকে রিভার্স অর্ডারে সাজানোর জাভা প্রোগ্রাম	২৯১
৭।	A থেকে Z পর্যন্ত বড় হাতের অক্ষরগুলো প্রদর্শনের জাভা প্রোগ্রাম	২৯১
৮।	break ও continue ব্যবহার করে দুটি সংখ্যার যোগফল নির্ণয়ের জাভা প্রোগ্রাম	২৯২
৯।	class ব্যবহার করে একটি জাভা প্রোগ্রাম	২৯৩
১০।	মেথড ব্যবহার করে একটি জাভা প্রোগ্রাম	২৯৪
১১।	মেথড ওভারলোডিং ব্যবহার করে একটি জাভা প্রোগ্রাম	২৯৬

১২। কনস্ট্রাক্টর ব্যবহার করে একটি জাভা প্রোগ্রাম ২৯৭
১৩। কপি কনস্ট্রাক্টর ব্যবহার করে একটি জাভা প্রোগ্রাম ২৯৭
১৪। ইনহেরিটেন্স বা সিঙ্গেল ইনহেরিটেন্স ব্যবহার করে একটি জাভা প্রোগ্রাম ২৯৯
১৫। মাল্টিলেভেল ইনহেরিটেন্স ব্যবহার করে একটি জাভা প্রোগ্রাম ৩০০
১৬। হায়ারারকিক্যাল ইনহেরিটেন্স ব্যবহার করে একটি জাভা প্রোগ্রাম ৩০২
১৭। ওভাররিডেন মেথড (Overridden Method) ব্যবহার করে একটি জাভা প্রোগ্রাম ৩০৩
১৮। অ্যাবস্ট্রাক্ট ক্লাস ব্যবহার করে একটি জাভা প্রোগ্রাম ৩০৪
১৯। ইন্টারফেস ব্যবহার করে একটি জাভা প্রোগ্রাম ৩০৫
২০। মাল্টিপল ইন্টারফেস ব্যবহার করে একটি জাভা প্রোগ্রাম ৩০৬
২১। মাল্টিথ্রেডেড পদ্ধতি ব্যবহার করে একটি জাভা প্রোগ্রাম ৩০৭
২২। getpriority ( ) ও setpriority method ব্যবহার করে একটি জাভা প্রোগ্রাম ৩০৮
২৩। থ্রেড সিনক্রোনাইজেশন বাস্তবায়নের একটি জাভা প্রোগ্রাম ৩১০

✪ নমুনা প্রশ্ন-২০১২ ..... ৩১২

সি ভাষায় গ্রাফিক্স
Graphics Elements in C

অধ্যায় ০১

১.০ ভূমিকা (Introduction) ঃ

কম্পিউটার মনিটরে অস্থায়ীভাবে প্রদর্শিত বর্ণ, সংখ্যা, বিশেষ ধরনের অক্ষর, ছবি, চিত্র ইত্যাদিকে কম্পিউটার গ্রাফিক্স (Computer graphics) বলে।

কম্পিউটার প্রোগ্রামিং এর ক্ষেত্রে গ্রাফিক্স একটি অতি গুরুত্বপূর্ণ বিষয়। কারণ কোন তথ্যের আউটপুট বা ফলাফল যদি চিত্র বা গ্রাফের মাধ্যমে প্রদর্শন করা যায় তাহলে তা অনেক বেশি অর্থবহ ও ফলদায়ক হয়। ইমেজ প্রসেসিং, মডেলিং, এনিমেশন, সিমুলেশন এপ্লিকেশন, ক্যাড (CAD-Computer Aided Design), ক্যাম (CAM-Computer Aided Manufacturing), ট্রেনিং ও প্রেজেন্টেশনে কম্পিউটার গ্রাফিক্স আজ বেশ জনপ্রিয়। C প্রোগ্রামিং ল্যাংগুয়েজের (বিশেষত ঃ Turbo C, Borland C ইত্যাদি কিন্তু ANSI C নয় কারণ ঃ ANSI C তে গ্রাফিক্স ফাংশন নেই) মাধ্যমে বিভিন্ন ধরনের গ্রাফিক্স প্রোগ্রামিং সহজে ও স্বচ্ছন্দে করা যায়।

C তে গ্রাফিক্স প্রোগ্রামিং করার জন্য কম্পিউটারে অবশ্যই Graphics video adapter সংযুক্ত থাকতে হবে। প্রচলিত কয়েকটি graphics adapter-এর মধ্যে MA (Monochrome Adapter), CGA (Color Graphics Adapter), VGA (Video Graphics Adapter) ইত্যাদি উল্লেখযোগ্য। এই অ্যাডাপ্টারগুলো একত্রে ব্যবহার করে ভিন্ন ভিন্ন মোডের নানান রকম টেক্সট ও গ্রাফিক্স অপারেশন সম্পন্ন করা যায়।

১.০.১ C প্রোগ্রামিং ল্যাংগুয়েজে গ্রাফিক্স প্রোগ্রামিং এর ক্ষেত্রে করণীয় ঃ

C তে গ্রাফিক্স প্রোগ্রামিং এর ক্ষেত্রে নিম্নবর্ণিত বিষয়াবলি বিবেচনায় রাখতে হয়–

- প্রথমেই <graphics.h> হেডার ফাইল include করতে হবে। কারণ <graphics.h> হেডার ফাইলে প্রায় 100 এর মত গ্রাফিক্স ফাংশন রয়েছে যার সাহায্য নিয়ে বিভিন্ন ধরনের গ্রাফিক্স প্রোগ্রামিং করা যায়। এজন্য-Turbo C এর বেলায় Menu থেকে options → libraries-এ গিয়ে Graphics library check box কে select করতে হবে। Borland C++ এর বেলায় New target icon এ click করে New target dialogue box এর platform Dos standard select করতে হবে।
  অতঃপর standard libraries এর BGI cneck box কে enable করে দিতে হবে।
- তারপর int টাইপের গ্রাফিক্স ড্রাইভার ও গ্রাফিক্স মুড ভেরিয়েবল ঘোষণা করতে হয়।
  যেমন– int driver, mode;
- অতঃপর গ্রাফিক্স ড্রাইভার ডিটেক্ট করে এর মান গ্রাফিক্স ড্রাইভার ভেরিয়েবলে রাখতে হয়।
  driver = DETECT;
  mode = 0;

সবশেষে initgraph ( ) ফাংশন ব্যবহার করে টেক্সট মুড থেকে গ্রাফিক্স মুডে পরিবর্তন করতে হবে। কারণ ঃ টেক্সট মুডে শুধুমাত্র ক্যারেক্টার প্রদর্শন করা যায় কিন্তু গ্রাফিক্স মুডে ক্যারেক্টার ও পিকচার উভয়ই প্রদর্শন করা যায়। টেক্সট মুডে স্ক্রিনের সর্ববামের উপরের (Left-top) বিন্দুর স্থানাঙ্ক (1, 1) অপরদিকে গ্রাফিক্স মুডের বেলায় এর স্থানাঙ্ক (0,0)।

উপরিউক্ত কাজগুলো ধারাবাহিকভাবে নিম্নলিখিত ধাপে সম্পন্ন হয়।

```
int driver, mode;
driver = DETECT;
mode = 0;
initgraph (& driver, & mode, "C : \\ Tc\\ bgi");
```

১.১ পিক্সেল ও রেজোলুশন (Concept of pixel and resolution) ঃ

Pixel এর পূর্ণরূপ হচ্ছে Picture Element। কম্পিউটার স্ক্রীনে আমরা যে ছবি বা চিত্র দেখি তা মূলত অসংখ্য ক্ষুদ্র ক্ষুদ্র ডট (Dot) বা আলোক বিন্দু নিয়ে গঠিত। এসকল ক্ষুদ্র ক্ষুদ্র ডট বা আলোক বিন্দুর প্রতিটিই এক একটি পিক্সেল (Pixel)। একটি ছবির পিক্সেল সংখ্যা যত বেশি হবে ছবিটি দেখতে তত বেশি বাস্তব প্রকৃতির হবে। অপরদিকে রেজোলুশন (Resolution) বলতে ছবির কোয়ালিটিকে বুঝায়। মনিটরের স্ক্রীনে হরাইজন্টাল ও ভার্টিক্যাল বরাবর মোট পিক্সেলের সংখ্যাকে রেজোলুশন বলে। প্রকৃতপক্ষে মনিটরের হরাইজন্টাল ও ভার্টিক্যাল বরাবর পিক্সেলের সংখ্যাই রেজোলুশনের পরিমাপ। যেমন- AGP কার্ডের রেজোলুশন 1024 × 768 এর অর্থ হল ঃ AGP card মনিটরের পর্দায় হরাইজন্টাল বরাবর 1024টি ও ভার্টিক্যাল বরাবর 768টি পিক্সেল সৃষ্টি করে। কম্পিউটার স্ক্রীনের কোন একক আয়তনে পিক্সেলের সংখ্যা যত বেশি হবে তার রেজোলুশনও তত বৃদ্ধি পায়। রেজোলুশন বৃদ্ধি পেলে ছবিটি দেখতে আরো বেশি স্বচ্ছ, সুস্পষ্ট, আকর্ষণীয়, জীবন্ত ও সমৃদ্ধ হয়ে উঠে। বিপরীতক্রমে রেজোলুশন কম হলে ছবিটি দেখতে ঝাপসা ও অস্পষ্ট প্রকৃতির হবে।

কম্পিউটার মনিটরের রেজোলুশন নির্ভর করে মূলত এর ভিডিও অ্যাডাপ্টারের উপর। আবার ভিডিও অ্যাডাপ্টারগুলোকে ও বিভিন্ন মুডে প্রোগ্রাম করা যায়। নিম্নে C/C++ এ ব্যবহৃত বিভিন্ন প্রকার গ্রাফিক্স অ্যাডাপ্টারের নাম, মুড ও রেজোলুশন উল্লেখ করা হল।

ড্রাইভার	মোড	রেজোলুশন
CGA	0	320 × 200
	1	320 × 200
	2	320 × 200
	3	320 × 200
	4	640 × 200
EGA	0	640 × 200
	1	640 × 350
VGA	0	640 × 200
	1	640 × 350
	2	640 × 480
IBM8514	0	640 × 480
	1	1024 × 768

টেবিল ঃ ১.১ ভিডিও মোড ও রেজোলুশন

১.২ উদাহরণসহ বিভিন্ন প্রকার গ্রাফিক্স ফাংশনের ফরম্যাট ও ব্যবহার (Format and use of line( ), rectangle( ), bar( ), bar3d( ), circle( ), ellipse(), fillellipse( ) and sector( ) Functions with Example.

line ( ) ফাংশন ঃ সরল রেখা অঙ্কনের জন্য line ( ) ফাংশনটি ব্যবহৃত হয়।

ফরম্যাট ঃ

line (startx, starty, endx, endy);

এখানে startx ও starty হল লাইন শুরুর স্থানাঙ্ক এবং endx ও endy হল লাইন এর শেষ স্থানাঙ্ক। স্থানাঙ্ক পিক্সেলে গণনা করা হয়।

উদাহরণ ঃ line (50, 50, 150, 150);

উপরোক্ত কমান্ডের মাধ্যমে (50, 50) স্থানাঙ্ক হতে (150, 150) স্থানাঙ্ক বা কো অর্ডিনেট পর্যন্ত সরল রেখা অঙ্কন করা যায়।

rectangle ( ) ফাংশন ঃ আয়তক্ষেত্র অঙ্কনের জন্য rectangle ( ) ফাংশন ব্যবহার করা হয়।

ফরম্যাট ঃ

rectangle (left, top, right, bottom);

এখানে left, top হচ্ছে আয়তক্ষেত্রের বামদিকের উপরের কৌণিক বিন্দুর স্থানাঙ্ক বা কো-অর্ডিনেট এবং right, bottom হচ্ছে ডানদিকের নিচের কৌণিক বিন্দুর স্থানাঙ্ক।

উদাহরণ ঃ ractangle (150, 90, 34, 300);

উল্লিখিত কমান্ডের মাধ্যমে (150, 90) left top স্থানাঙ্ক ও (34, 300) right, bottom স্থানাংকের একটি আয়তক্ষেত্র অঙ্কন করা যাবে।

bar ( ) ফাংশন ঃ Rectangular Bar অঙ্কনের জন্য bar ( ) ফাংশনটি ব্যবহৃত হয়।

ফরম্যাট ঃ

bar (left, top, right, bottom);

left, top হল Bar এর বামদিকের উপরের কৌণিক বিন্দুর স্থানাঙ্ক মান এবং right, bottom হল Bar এর ডানদিকের নিচের কৌণিক বিন্দুর স্থানাঙ্ক মান।

উদাহরণ ঃ bar, (110, 110, 130, 210);

উপরোক্ত স্টেটমেন্টের মাধ্যমে (110, 110) top left স্থানাঙ্ক ও (130, 210) right, bottom স্থানাঙ্কের একটি Bar অঙ্কন করা যায়।

bar3d ( ) ফাংশন ঃ ত্রিমাত্রিক (Three dimensional) বার অঙ্কনের জন্য bar3d ( ) ফাংশন ব্যবহৃত হয়।

ফরম্যাট ঃ

bar3d (left, top, right, bottom, depth, topflag);

যেখানে, left, top ত্রিমাত্রিক বারের বামদিকের উপরের স্থানাঙ্ক মান, right, bottom ত্রিমাত্রিক বারের ডানদিকের নিচের স্থানাঙ্ক মান, depth ত্রিমাত্রিক বারের পুরুত্ব এবং topflag ত্রিমাত্রিক বারের closing top এর সংখ্যা মানকে নির্দেশ করে।

উদাহরণ ঃ bar3d (200, 100, 220, 200, 10, 1);

উল্লিখিত কমান্ড ব্যবহার করে নির্দেশিত স্থানাঙ্ক অনুসারে একটি ত্রিমাত্রিক বার অঙ্কন করা যায়।

circle ( ) ফাংশন ঃ circle ( ) ফাংশন ব্যবহার করে বৃত্ত অঙ্কন করা যায়।

ফরম্যাট ঃ

circle (x, y, radius);

এখানে (x, y) হল বৃত্তের কেন্দ্রের স্থানাঙ্ক মান এবং madius হল বৃত্তের ব্যাসার্ধের সংখ্যা মান।

উদাহরণ ঃ circle (400, 200, 100); উপরোক্ত কমান্ডের মাধ্যমে প্রদেয় কেন্দ্র স্থানাঙ্ক (400, 200) এবং ব্যাসার্ধ (100) ব্যবহার করে একটি বৃত্ত অঙ্কনের নির্দেশ প্রদান করা হয়েছে।

ellipse ( ) ফাংশন ঃ ellipse ( ) ফাংশনের মাধ্যমে উপবৃত্ত অঙ্কন করা যায়।

ফরম্যাট ঃ

ellipse (x, y, start, end, x radius, y radius);

এখানে, (x, y) হল উপবৃত্তের কেন্দ্রের স্থানাঙ্ক মান, start ও end হল উপবৃত্তের রেডিয়ান কোণের মান যা মূলত start এর জন্য 0° ডিগ্রি এবং end এর জন্য 360 ডিগ্রি হয়ে থাকে। কারণ stard ও end এর রেডিয়ান কোণের মান যথাক্রমে 0 ডিগ্রি এবং 360 ডিগ্রি হলেই উপবৃত্তের পুরো অংশ দৃশ্যমান হবে। আর x radius ও y radius হল যথাক্রমে x অক্ষের ব্যাসার্ধ মান এবং y অক্ষের ব্যাসার্ধ মান।

উদাহরণ ঃ ellipse (100, 100, 0, 360, 80, 40);

প্রদেয় কমান্ডটির মাধ্যমে উল্লিখিত স্থানাঙ্ক, রেডিয়ান কোণ ও ব্যাসার্ধসহ উপবৃত্ত/অধিবৃত্ত অঙ্কনের নির্দেশ দেয়া হয়েছে।

fillellipse ( ) ফাংশন ঃ বর্তমানে সক্রিয় ও ফিল স্টাইলে (Fill style) উপবৃত্ত অঙ্কন ও ভরাট (fill) করার জন্য fillelipse ( ) ফাংশনটি ব্যবহৃত হয়।

ফরম্যাট ঃ

fillellipse (x, y, xradius, yradius);

sector ( ) ফাংশন ঃ sector ( ) ফাংশনটি উপবৃত্তের এরিয়াকে বর্তমানে সক্রিয় রঙ ও ফিল প্যাটার্ন (Fill pattern) দ্বারা পূর্ণ (Fill) করে।

ফরম্যাট ঃ

sector (x, y, start, end, xradius, yradius);

এখানে, (x, y) হল উপবৃত্তের কেন্দ্রের স্থানাঙ্ক, (start ও end) হল রেডিয়ান কোণের মান এবং (x radius এবং y radius) হল x অক্ষ ও y অক্ষের ব্যাসার্ধ।

উদাহরণ ঃ sector (300, 200, 0, 360, 100, 200);

১.৩ ফরম্যাট, ব্যবহার ও উদাহরণসহ বিভিন্ন ধরনের ফাংশন (Format and use of arc ( ), pieslice( ), drawpoly( ), fillPoly( ), outtextxy( ), settextstyle( ), cleardevice( ), delay( ), sound( ) and nosound( ) Functions with Example) ঃ

arc ( ) ফাংশন ঃ বৃত্তাংশ বা বৃত্তচাপ অঙ্কনের জন্য arc ( ) ফাংশন ব্যবহার করা হয়।

ফরম্যাট ঃ

arc (x, y, start, end, radius);

এখানে, (x, y) হল বৃত্তের কেন্দ্রের স্থানাঙ্ক মান, radius হল ব্যাসার্ধ এবং start ও end দ্বারা যথাক্রমে বৃত্তাংশ দ্বারা তৈরি বৃত্তের কেন্দ্রে তৈরি কোণের শুরু ও শেষ বুঝায়। কোণকে ডিগ্রিতে প্রকাশ করা হয়।

উদাহরণ ঃ arc (200, 150, 0, 90, 50);

উপরোক্ত কমান্ডের ফলে প্রদত্ত স্থানাঙ্ক মান (200, 150), ব্যাসার্ধ মান (50) এবং কৌণিক মান (0, 90) এর উপর ভিত্তি করে একটি বৃত্তচাপ অঙ্কিত হবে।

pieslice ( ) ফাংশন ঃ pieslice ফাংশনটি বর্তমানে সক্রিয় রঙ ও প্যাটার্ন এর pieslice অঙ্কনের জন্য ব্যবহৃত হয়।

ফরম্যাট ঃ

```
pieslice (x, y start, end, radius);
```

এখানে, (x, y) হল pieslice এর কেন্দ্র বিন্দুর স্থানাঙ্ক, (start ও end) হল pieslice এর শুরু ও শেষের কৌনিক মান (ডিগ্রিতে) এবং radius হল ব্যাসার্ধ মান।

উদাহরণ ঃ pieslice (300, 200, 0, 60, 50);

drawpoly ( ) ফাংশন ঃ বহুভুজ (Polygon) অঙ্কনের জন্য

drawpoly ( ) ফাংশনটি ব্যবহৃত হয়।

ফরম্যাট ঃ

```
drawpoly (numpoints, polypoints);
```

numpoints দ্বারা বহুভুজের পয়েন্ট সংখ্যা নির্ধারণ করা হয়। অর্থাৎ চতুর্ভুজ হলে numpoints এর মান হবে 4, পঞ্চভুজ হলে হবে 5 কিংবা ষড়ভুজ হলে হবে 6। আর Poly points দ্বারা বহুভুজের অ্যারের নাম নির্দেশ করে যেখানে অ্যারে সাইজ বহুসংখ্যার দ্বিগুণ।

উদাহরণ ঃ পঞ্চভুজ, আঁকার জন্য-

```
int shape [10] = {15, 15, 50, 50, 90, 100, 50, 150, 0, 0];
drawpoly (5, shape);
```

উপরোক্ত statement সমূহের মাধ্যমে 5টি পয়েন্ট ও প্রতিটি পয়েন্টের কো-অর্ডিনেট (15, 15, 50, 50, 90, 100, 50, 150, 0, 0) যুক্ত একটি পঞ্চভুজ তৈরি হবে। আর কো-অর্ডিনেট পয়েন্টসমূহ সংরক্ষণের জন্য Shape নামক অ্যারে ব্যবহার করা হয়েছে।

fillpoly ( ) ফাংশন ঃ বর্তমানে সক্রিয় রঙ ও প্যাটার্ন ব্যবহার করে একটি fill করা বহুভুজ অঙ্কন করতে fillpoly ( ) ফাংশনটি ব্যবহৃত হয়।

ফরম্যাট ঃ

```
fillpoly (numpoints, polypoints);
```

উদাহরণ ঃ

```
int shape [10] = {15, 15, 50, 50, 90, 100, 50, 150, 0, 0};
drowpoly (5, shape);
fillpoly (5, shape);
```

outtextxy ( ) ফাংশন ঃ বর্তমানে সক্রিয় ফন্ট, ফন্ট সাইজ ও ডিরেকশন অনুযায়ী কোন টেক্সট স্ট্রিং (text string) কে ভিউপোর্টের (মনিটরের প্রর্দায় যে অংশের মধ্যে গ্রাফিক্স ফাংশন কাজ করে) নির্দিষ্ট পজিশনে ডিসপ্লে (Display) করার জন্য outtextxy ( ) ফাংশনটি ব্যবহৃত হয়।

ফরম্যাট ঃ

```
outtextxy (x, y, string);
```

এখানে, (x, y) হল ভিউপোর্টের পজিশন স্থানাঙ্ক এবং যেই টেক্সটকে ডিসপ্লে করতে চাই তাই হল string.

উদাহরণ ঃ outtextxy (40, 290, "Programming language-2"),

উপরোক্ত statement এর মাধ্যমে "Programming language-2" stringটি ভিউপোর্টের (40, 290) পজিশনে ডিসপ্লে হবে।

settextstyle ( ) ফাংশন ঃ settextstyle ( ) ফাংশনটি textstyle অর্থাৎ টেক্সট এর ফন্ট, ডিরেকশন ও ফন্ট সাইজ ইত্যাদি সেট করার কাজে ব্যবহৃত হয়।

ফরম্যাট ঃ

```
settextstyle (font, direction, size);
```

এখানে 'font' দ্বারা font name, 'direction' দ্বারা হরাইজন্টাল বা ভার্টিক্যাল ডিরেকশন এবং 'size' দ্বারা ফন্ট সাইজকে বুঝানো হয়েছে। উল্লেখ্য, <graphics. h> হেডার ফাইলে font name এর জন্য 5টি ম্যাক্রো, Direction এর জন্য 2টি macro ব্যবহৃত হয় যা নিম্নে টেবিল আকারে উল্লেখ করা হল। আর font size এর মান 1 থেকে 10 পর্যন্ত ব্যবহৃত হয়।

ফন্টের নাম	সমতুল্য মান
DEFAULT-FONT	0
TRIPLEX – FONT	1
SMALL-FONT	2
SANS-SERIF-FONT	3
GOTHIC-FONT	4

টেবিল ঃ ফন্ট স্টাইল ও তাদের সমতুল্য মান

ডিরেকশন	সমতুল্য মান
HORIZ-DIR	0
VERT-DIR	1

টেবিল ঃ ডিরেকশন ও সমতুল্যমান

উদাহরণ ঃ

```
settextstyle (GOTHIC-FONT, VERT-DIR, 4);
outtext ("TAHSIN");
```

উপরোক্ত স্টেটমেন্টদ্বয়ের জন্য "TAHSIN" লাইনটি উল্লম্বভাবে স্বাভাবিকের চেয়ে 4 গুণ বড় আকারে প্রিন্ট হবে।

cleardevice ( ) ফাংশন ঃ টেক্সট মোডে স্ক্রীনকে পরিষ্কার করার কাজে যেমন clrscr ( ) ফাংশনটি ব্যবহৃত হয় তদ্রূপ গ্রাফিক্স মোডে স্ক্রীনকে পরিষ্কার করার কাজে cleardevice ( ) ফাংশনটি ব্যবহৃত হয়। এর ফলে স্ক্রীনের বর্তমান পজিশন হয় (0, 0)।

ফরম্যাট ঃ

```
cleardevice ( );
```

delay ( ) ফাংশন ঃ delay ( ) ফাংশনটি প্রোগ্রাম অপারেশনকে কিছু নির্দিষ্ট সময়ের (মিলি সেকেন্ড) জন্য বিরত রাখে। এই ফাংশনটির জন্য <dos. h> হেডার ফাইলটি include করতে হবে।

ফরম্যাট ঃ

```
delay (mill sec);
```

এখানে millsec দ্বারা সময়ের পরিমাণকে মিলি সেকেন্ডে argument হিসেবে সেট করতে হয়।

উদাহরণ ঃ delay (700);

উপরোক্ত স্টেটমেন্টের জন্য প্রোগ্রাম অপারেশন 700 মিলি সেকেন্ড সময়ের জন্য delay হবে।

sound ( ) ফাংশন ঃ একটি নির্দিষ্ট ফ্রিকুয়েন্সিতে কম্পিউটারের স্পিকারে সাউন্ড শুরু করতে sound ( ) ফাংশনটি ব্যবহৃত হয়। এর হেডার ফাইল ও <dos.h>

ফরম্যাট ঃ

```
sound (hertz);
```

এখানে hertz বলতে নির্দিষ্ট ফ্রিকুয়েন্সিকে বুঝানো হয়েছে। সাধারণত ফ্রিকুয়েন্সি 20-20000 hertz এর মধ্যে হতে হবে। নচেৎ শব্দ শোনা যাবে না।

উদাহরণ ঃ

```
sound (100);
```

উক্ত স্টেটমেন্টের কারণে 100 hertz এর শব্দ শোনা যাবে।

nosound ( ) ফাংশন ঃ sound ( ) ফাংশন এর কারণে শুরু কোন শব্দকে বন্ধ করতে nosound ( ) ফাংশন ব্যবহৃত হয়। এর হেডার ফাইল <dos.h>

ফরম্যাট ঃ

```
nosound ( );
```

setcolor ( ) ফাংশন ঃ setcolor ( ) ফাংশনটি ড্রইং কালার নির্ধারণের জন্য ব্যবহৃত হয়। ফাংশনে বর্ণিত ম্যাক্রো বা তার সমতুল্য মান অনুযায়ী ড্রইং কালার নির্ধারিত হয়।

ফরম্যাট ঃ

```
setcolor (color);
```

এখানে color একটি ইন্টিজার সংখ্যা বা ম্যাক্রো যা বিশেষ কোন রঙকে নির্দেশ করে।

উদাহরণ ঃ

```
setcolor (4);
circle (300, 200, 50);
```

অথবা

```
setcolor (RED);
circle (300, 200, 50)
```

উপরোক্ত দুটো ক্ষেত্রেই একটি লাল রঙের বৃত্ত অঙ্কিত হবে।

setbkcolor ( ) ফাংশন ঃ ব্যাক গ্রাউন্ড কালার পরিবর্তনের জন্য setbkcolor ( ) ফাংশনটি ব্যবহৃত হয়। ফাংশনে বর্ণিত ম্যাক্রো বা তার সমতুল্য মান অনুযায়ী ব্যাকগ্রাডিন্ডের কালার পরিবর্তন হয়।

ফরম্যাট ঃ

```
setbkcolor (color);
```

যেখানে Color একটি ইন্টিজার সংখ্যা বা ম্যাক্রো যা বিশেষ কোন রঙকে নির্দেশ করে।

উদাহরণ ঃ

```
setbkcolor (BLUE);
setcolor (RED);
circle (300, 200, 50);
```

অথবা

```
setbkcolor (1);
setcolor (4);
circle (300, 200, 50);
```

উপরোক্ত উভয় ক্ষেত্রে নীল রঙের ব্যাকগ্রাউন্ডে একটি লাল রঙের বৃত্ত অঙ্কিত হবে।

নিম্নে বিভিন্ন রঙের ম্যাক্রো ও কোড নম্বর একটি টেবিলের মাধ্যমে উপস্থাপন করা হল।

নির্দিষ্ট কোন রঙের নির্দিষ্ট কোন প্যাটার্ন দিয়ে পূর্ণ করার জন্য।

ম্যাক্রো	ইন্টিজার সংখ্যা	রঙ
BLACK	0	কালো
BLUE	1	আকাশী নীল
GREEN	2	সবুজ
CYAN	3	নীল
RED	4	লাল
MAGENTA	5	রক্ত লাল
BROWN	6	বাদামি
LIGHTGRAY	7	হালকা ধূসর
DARKGRAY	8	গাঢ় ধূসর
LIGHTBLUE	9	হালকা নীল
LIGHTGREEN	10	হালকা সবুজ
LIGHTCYAN	11	হালকা বেগুনি
LIGHTRED	12	হালকা লাল
LIGHTMAGENTA	13	হালকা রক্ত বর্ণ
YELLOW	14	হলুদ
WHITE	15	সাদা

টেবিল ঃ রঙের ম্যাক্রো

setfillstyle ( ) ফাংশন ঃ কোন ড্রইং অবজেক্টের ফিলিং স্টাইল অর্থাৎ কোন ড্রইং অবজেক্টকে নির্দিষ্ট কোন রঙের নির্দিষ্ট কোন প্যাটার্ন দ্বারা পূর্ণ করার জন্য setfillstyle ( ) ফাংশন ব্যবহৃত হয়।

ফরম্যাট ঃ

```
setfillstyle (pattern, color);
```

এখানে pattern ও color উভয়ই হচ্ছে একটি ইন্টিজার সংখ্যা বা ম্যাক্রো।

উদাহরণ ঃ

```
circle (300, 200, 50);
setfillstyle (LINE-FILL, 4);
```

উল্লিখিত উদাহরণের ক্ষেত্রে দেখা যাবে যে একটি বৃত্তে লাল রঙের লাইন দ্বারা পূর্ণ করা হয়েছে। নিম্নে প্যাটার্নের জন্য নির্ধারিত ম্যাক্রোসমূহের তালিকা উল্লেখ করা হল।

ম্যাক্রো	ইন্টিজার সংখ্যা	বর্ণনা
EMPTY_FILL	0	ব্যাকগ্রাউন্ড রঙে পূর্ণ করে
SOLID_FILL	1	সলিড রঙে পূর্ণ করে
LINE_FILL	2	লাইন দ্বারা পূর্ণ করে
LTSLASH_FILL	3	হালকা স্ল্যাশ দ্বারা পূর্ণ করে
BKSLASH_FILL	4	স্ল্যাশ দ্বারা পূর্ণ করে
BKSLASH_FILL	5	ব্যাক স্ল্যাশ দ্বারা পূর্ণ করে
LTBKSLASH_FILL	6	হালকা ব্যাক স্ল্যাশ দ্বারা পূর্ণ করে
HATCH_FILL	7	হালকা হ্যাচ দ্বারা পূর্ণ করে
XHATCH_FILL	8	হ্যাচ দ্বারা পূর্ণ করে
INTERLEAVE_FILL	9	ইন্টারলিভ দ্বারা পূর্ণ করে
WIDE_DOT_FILL	10	অল্প সংখ্যক ডট দ্বারা পূর্ণ করে
CLOSE_DOT_FILL	11	বহু সংখ্যক ডট দ্বারা পূর্ণ করে
USER_FILL	12	নিজস্ব প্যাটার্ন দ্বারা পূর্ণ করে

টেবিল ঃ রঙ দিয়ে পূর্ণ করার স্টাইলের কোড এবং এর ম্যাক্রো

floodfill ( ) ফাংশন ঃ কোন আবদ্ধ ক্ষেত্র যেমন- চতুর্ভুজ, বৃত্ত, বহুভুজ, রেখা দ্বারা আবদ্ধ ক্ষেত্র ইত্যাদিকে নির্ধারিত রঙে পূর্ণ করার জন্য floodfill ( ) ফাংশনটি ব্যবহৃত হয়।

ফরম্যাট ঃ

```
floodfill (x, y, bordercolor);
```

এখানে, (x, y) হল কোন আবদ্ধ ক্ষেত্রের কোন বিন্দুর স্থানাঙ্ক এবং bordercolor হল উক্ত ক্ষেত্রে চারপাশের বর্ডার কালার।

উদাহরণ ঃ

```
circle (300, 200, 50);
floodfill (300, 200, GREEN);
```

উপরোক্ত কমান্ডটি (300, 200) স্থানাঙ্ক বিশিষ্ট আবদ্ধ ক্ষেত্র একটি বৃত্তের চারপাশকে সবুজ দেখাবে।

১.৪ ইনপুট আউটপুট অপারেশনের ক্ষেত্রে cprintf ( ) ও cscanf ( ) ফাংশনের ব্যবহার (Use of modified cprintf ( ) and cscanf ( ) functions for I/O operation)

ইনপুট আউটপুট অপারেশনের ক্ষেত্রে scanf ( ) ও printf ( ) এ দু'টি ফাংশন ব্যবহৃত হয়। cscanf ( ) ও cprintf ( ) ফাংশন দুটি ও ইনপুট-আউটপুট অপারেশনেই ব্যবহৃত হয়। তবে পার্থক্য এই যে, মনিটর স্ক্রীনে ইনপুট-আউটপুটের জন্য যথাক্রমে scanf ( ) ও printf ( ) ফাংশন কাজ করে। অপরদিকে csanf ( ) ও cprintf ( ) ফাংশন দুটির ইনপুট-আউটপুট অপারেশন সম্পন্ন হয় সক্রিয় (active) উইন্ডোতে। নিম্নে উদাহরণসহ cprintf ( ) ফাংশন ও cscanf ( ) ফাংশন উল্লেখ করা হল।

cprintf () ফাংশন ঃ এ ফাংশনটির কাজ printf() এর মতই। পার্থক্য হল cprintf () প্রিন্ট করে সক্রিয় উইন্ডোতে। এ ফাংশনে স্ল্যাশ (\n) কোড কার্সর কে পরের লাইনে স্থাপন করে, কিন্তু পরের লাইনের শুরুতে নেয় না। পরের লাইনের শুরুতে কার্সর স্থাপন করার জন্য (\n\r) কোড নিতে হয়। নিচের প্রোগ্রামটি একটি উইন্ডোর ভিতর 20, 2 অবস্থানে This is a test. স্ট্রিং প্রিন্ট করে এবং কার্সর পরের লাইনের শুরুতে যায়।

```
#include <graphics.h>
#include <stdio.h>
#include <conio.h>
void main ()
{ int driver, mode;
driver = DETECT;
mode = 0;
initgraph (&driver, &mode, "c:\\tc\\bgi");
textbackground (0);
clrscr();
window (10, 5, 69, 20);
textbackground(CYAN);
clrscr();
gotoxy (20, 2);
cprintf("This is a test. \n\r");
getch ();
}
```

cscanf () ফাংশন ঃ উইন্ডোর ভিতর কনসোল হতে ইনপুট গ্রহণ করার জন্য এ ফাংশন ব্যবহার করা হয়। নিচের প্রোগ্রামে উইন্ডোর ভিতর হতে নাম ও বয়স গ্রহণ করবে।

```
#include <graphics.h>
#include <stdio.h>
#include <conio.h>
void main ()
{ int driver, mode;
driver = DETECT;
mode = 0;
initgraph (&driver, &mode, "c:\\tc\\bgi");
char names [20];
int age;
textbackground (0);
clrscr();
window (10, 5, 69, 20);
textbackground(RED);
clrscr();
cprintf("Enter your name :");
cscanf ("%s", names);
cprintf ("\n\rEnter age :");
cscanf ("%d", &age);
getch ();
}
```

১.৫ গ্রাফিক্স ফাংশন ব্যবহার করে কালার ইমেজ তৈরির প্রোগ্রাম (Developing color image using mentioned graphics functions) ঃ

প্রোগ্রাম-১৷ আয়তক্ষেত্র অংকণের প্রোগ্রাম ঃ

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<graphics.h>
void main()
{
int driver,mode;
driver=DETECT;
mode=0;
initgraph(&driver,&mode,"c:\\tc\\bgi");
rectangle(150,90,34,300);
getch();
}
```

প্রোগ্রাম-২। বৃত্ত অংকণের সি প্রোগ্রাম ঃ

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<graphics.h>
void main()
{
int driver,mode;
driver=DETECT;
mode=0;
initgraph(&driver,&mode,"c:\\tc\\bgi");
circle(300,300,50);
getch();
}
```

প্রোগ্রাম-৩। উপবৃত্ত (ellipse) অংকণের প্রোগ্রাম ঃ

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<graphics.h>
void main()
{
int driver,mode;
driver=DETECT;
mode=0;
initgraph(&driver,&mode,"c:\\tc\\bgi");
ellipse(100,100,0,360,80,40);
getch();
}
```

প্রোগ্রাম-৪। বার (bar) অংকণের সি প্রোগ্রাম ঃ

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
void main()
{
int driver, mode;
driver=DETECT;
mode=0;
initgraph(&driver,&mode,"c:\\tc\\bgi");
bar(100,100,120,200);
getch();
}
```

প্রোগ্রাম-৫। 3D বার অংকণের সি প্রোগ্রাম ঃ

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<graphics.h>
void main()
{
int driver,mode;
driver=DETECT;
mode=0;
initgraph(&driver,&mode,"c:\\tc\\bgi");
bar3d(200,100,220,200,10,1);
getch();
}
```

প্রোগ্রাম-৬। বহুভুজ অংকণের প্রোগ্রাম ঃ

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<graphics.h>
void main()
{
int driver,mode;
int shape[12]={0,150,100,50,200,50,300,150,200,250,100,250};
driver=DETECT;
mode=0;
initgraph(&driver,&mode,"c:\\tc\\bgi");
drawpoly(6,shape);
getch();
}
```

প্রোগ্রাম-৭। ত্রিমাত্রিক অবজেক্ট (ডিম) অংকণের প্রোগ্রাম ঃ

```
#include<stdio.h>
#include<conio.h>
#include<dos.h>
#include<graphics.h>
void main()
{
int driver,mode;
driver=DETECT;
initgraph(&driver,&mode,"c:\\tc\\bgi");
ellipse(100,100,0,360,80,40);
floodfill(100,100,WHITE);
outtextxy(40,50,"Egg");
getch();
closegraph();
}
```

প্রোগ্রাম-৮৷ গ্রাফিক্স মোডে টেক্সট (outtext) প্রদর্শনের প্রোগ্রাম ঃ

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<graphics.h>
void main()
{
int driver,mode;
driver=DETECT;
mode=0;
initgraph(&driver,&mode,"c:\\tc\\bgi");
outtext("Welcome to Computer Department");
outtextxy(50,100,"Welcome to Computer Department");
getch();
}
```

প্রোগ্রাম-৯৷ বাংলাদেশের জাতীয় পতাকা অংকণের প্রোগ্রাম ঃ

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<graphics.h>
void main()
{
int driver,mode;
driver=DETECT;
mode=0;
initgraph(&driver,&mode,"c:\\tc\\bgi");
setcolor(WHITE);
circle(320,240,100);
setfillstyle(1,RED);
floodfill(320,240,WHITE);
setcolor(WHITE);
rectangle(40,80,600,400);
setfillstyle(1,GREEN);
floodfill(320,240,WHITE);
setcolor(WHITE);
line(38,80,38,479);
setfillstyle(1,GREEN);
floodfill(320,240,WHITE);
getch();
}
```

১.৬ সি প্রোগ্রামিং ল্যাংগুয়েজে ইমেজ সংরক্ষণ ও ইমেজ লোডিং পদ্ধতি (Procedure of saving and loading an image in C) ঃ

বিভিন্ন ধরনের গ্রাফিক্স ফাংশন ব্যবহার করে ইমেজ তৈরি (Develop) করার পর ডেভেলপকৃত এসব ইমেজকে পরবর্তীকালে ব্যবহারের জন্য ইমেজকে সংরক্ষণ ও সংরক্ষিত ইমেজকে আবার প্রয়োজনে লোড করা খুবই গুরুত্বপূর্ণ ব্যাপার।

ডেভেলপকৃত ইমেজকে সংরক্ষণের জন্য getimage ( ) ফাংশন এবং সংরক্ষিত ইমেজকে প্রয়োজনে পর্দায় লোড করার জন্য putimage ( ) ফাংশন ব্যবহৃত হয়। নিম্নে getimage ( ) ফাংশন ও putimage ( ) ফাংশনের বর্ণনা দেয়া হল।

getimage ( ) ফাংশন ঃ মনিটরের পর্দায় প্রদর্শিত ইমেজ বা ছবিকে মেমোরিতে সংরক্ষণ (save/copy) করার জন্য getimage ( ) ফাংশনটি ব্যবহৃত হয়।

getimage ( ) ফাংশনটির ফরম্যাট নিম্নরূপ ঃ

```
getimage (left, top, right, bottom, buf);
```

যেখানে, left, top, right ও bottom স্থানাঙ্কের কৌণিক বিন্দু দ্বারা চতুর্ভুজ আকৃতির কোন ছবিকে নির্দেশ করে। আর 'buf' হল ছবিটি মেমোরির যে অংশে কপি বা সেভ হবে তার অ্যাড্রেস নির্দেশকারী একটি পয়েন্টার।

putimage ( ) ফাংশন ঃ getimage ( ) ফাংশন ব্যবহার করে মেমোরিতে সংরক্ষিত কোন ইমেজ বা ছবিকে মনিটরের পর্দায় পুনরায় লোড করার জন্য putimage ( ) ফাংশনটি ব্যবহৃত হয়।

putimage ( ) ফাংশনের ফরম্যাট নিরূপ ঃ

```
putimage (x, y, buf, op);
```

এখানে, (x, y) হল ছবিটির বামদিকের উপরের কৌণিক বিন্দুর স্থানাঙ্ক মান স্ক্রীনের যেখানে ইমেজ বা ছবিটি লোড বা প্রদর্শিত হবে। ছবিটির পর্দার রঙ এবং ছবির রঙ মিলিতভাবে কেমন দেখাবে তা নির্ভর করে 'op' এর উপর। নিম্নে 'op' এবং এর ম্যাক্রো তালিকা উল্লেখ করা হল।

ম্যাক্রো	ইন্টেজার সংখ্যা	বর্ণনা
COPY_PUT	0	কপিকৃত ছবির ন্যায় হুবহু প্রদর্শিত হবে।
XOR_PUT	1	প্রদর্শিত স্থানের সাথে কপিকৃত ছবি XOR করে প্রদর্শিত হবে।
OR_PUT	2	প্রদর্শিত স্থানের সাথে কপিকৃত ছবি OR করে প্রদর্শিত হবে।
AND_PUT	3	প্রদর্শিত স্থানের সাথে কপিকৃত ছবি AND করে প্রদর্শিত হবে।
NOT_PUT	4	কপিকৃত ছবি NOT করে প্রদর্শিত হবে।

টেবিল ঃ ছবি কপি করার অপশনের কোড এবং এর ম্যাক্রো

প্রোগ্রাম ঃ getimage ( ) ও putimage ( ) ফাংশন ব্যবহার করে একটি প্রোগ্রাম নিম্নরূপ ঃ

```
#include <graphics.h>
#include <conio.h>
#include <stdio.h>
#include <dos.h>
void main()
{
int driver,mode,x1,y1,n1,m1,incx,incy;
unsigned size;
```

```
void *buf;
driver=DETECT;
mode=0;
initgraph(&driver,&mode,"c:\\tc\\bgi");
x1=getmaxx();
y1=getmaxy();
setcolor(CYAN);
incx=10;incy=10;
circle(10,10,10);
setfillstyle(SOLID_FILL,CYAN);
floodfill(10,10,CYAN);
size=imagesize(0,0,20,20);
buf=malloc(size);
getimage(0,0,20,20,buf);
n1=0;
m1=0;
for(;;){
putimage(n1,m1,buf,XOR_PUT);
n1=n1+incx;
m1=m2+incy;
if(n1<0){
n1=10;
incx=-incx;
}
if(n1>x1){
n1=x1-10;
incx=incx;
}
if(m1<0){
m1=10;
incy=-incy;
}
if(m1>y1){
m1=y1-10;
incy=incy;
}
putimage(n1,m1,buf,COPY_PUT);
delay(500);
}
restorecrtmode();
}
```

১.৭ টেক্সট স্ট্রিংকে স্থান পরিবর্তন করার প্রক্রিয়া (Procedure to move text string on the screen) ঃ

কোন টেক্সট স্ট্রিংকে স্ক্রীনের এক জায়গা থেকে অন্য জায়গায় স্থানান্তরের জন্য movetext ( ) ফাংশনটি ব্যবহৃত হয়। movetext ( ) ফাংশনের ফরম্যাট হল ঃ

```
movetext (left, top, right, bottom, newleft, newtop);
```

এখানে left, top, right, bottom স্থানাঙ্কের কৌণিক বিন্দু দ্বারা একটি rectangle area কে বুঝানো হয় যার মধ্যে আবদ্ধ টেক্সটকে অন্য জায়গায় কপি বা মুভ করানো হবে। আর newleft, newtop দ্বারা সেই স্থানাঙ্ককে বুঝানো হয় যেখানে টেক্সটি মুভ হবে। অর্থাৎ নতুন জায়গার অ্যাড্রেস নির্ধারণের জন্যই newleft ও newtop ব্যবহৃত হয়।

টেক্সট স্ট্রিংকে মনিটরের পর্দায় স্থান পরিবর্তনের একটি প্রোগ্রাম নিম্নরূপ ঃ

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<graphics.h>
#include<dos.h>
void main()
{int x,y;
clrscr();
x=1;
y=1;
for(;;)
{
gotoxy(x,y);
puts("Welcome to CPI");
movetext(x,y,x+20,y+1,x+1,y+1);
gotoxy(x,y);
clreol();
x=x+1;
y=y+1;
if(x<59)
x=1;
if(y<24)
y=1;
delay(700);
}
getch();
}
```

অনুশীলনী-১

অতি সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। কম্পিউটার গ্রাফিক্স কাকে বলে?

উত্তরঃ কম্পিউটার মনিটরে অস্থায়ীভাবে প্রদর্শিত বর্ণ, সংখ্যা, বিশেষ ধরনের অক্ষর, ছবি, চিত্র ইত্যাদিকে কম্পিউটার গ্রাফিক্স বলে।

২। গ্রাফিক্সের লাইব্রেরি ফাংশন কী?

উত্তরঃ # include<graphics.h>

৩। Pixel কাকে বলে?

উত্তরঃ কম্পিউটারের স্ক্রীনে আমরা যে ছবি দেখি তা অসংখ্য ক্ষুদ্র ক্ষুদ্র আলোক বিন্দু নিয়ে গঠিত। এসব আলোক বিন্দুর প্রতিটিই এক একটি pixel.

৪। রেজোলুশন (Resolution) বলতে কী বুঝ?

উত্তরঃ মনিটরের পর্দায় হরাইজন্টাল ও ভার্টিক্যাল বরাবর মোট পিক্সেলের সংখ্যাকে রেজোলুশন বলে। প্রকৃতপক্ষে মনিটরের অনুভূমিক ও উল্লম্ব বরাবর পিক্সেলের সংখ্যাই রেজোলুশনের পরিমাপ।

৫। graphics driver কী? এরা কোথায় সংরক্ষিত থাকে?

উত্তরঃ Graphics driver এমন এক ধরনের program যা graphics card বা adapter কে সক্রিয় করে তোলে। এই driver গুলো সাধারণত BGI ফাইলে সংরক্ষিত থাকে।

৬। initgraph () ফাংশনের কাজ কী?

উত্তরঃ ভিডিও অ্যাডাপ্টারকে গ্রাফিক্স মোডে সক্রিয় করার জন্য initgraph () ব্যবহার করা হয়।

৭। টেক্সট (Text) ও গ্রাফিক্স (Graphics) মোডের মধ্যে তফাত কী?

উত্তরঃ Text mode এ শুধুমাত্র ক্যারেক্টার প্রদর্শন করা যায়, কিন্তু Graphics মোডে character ও graphics image উভয়ই প্রদর্শন করা যায়।

৮। প্রচলিত কয়েকটি ডিসপ্লে অ্যাডাপ্টারের নাম লিখ।

উত্তরঃ প্রচলিত কয়েকটি Graphics adapter এর মধ্যে MA (Monochrome Adapter), CGA (Colour Graphics Adapter), VGA (Video Graphics Adapter) ইত্যাদি উল্লেখযোগ্য।

৯। restorecrtmode () ফাংশনের কাজ কী?

উত্তরঃ গ্রাফিক্স মোড থেকে প্রোগ্রাম শেষে অপারেটিং সিস্টেমে ফিরে আসার পূর্বে স্ক্রীন মোড পুনঃস্থাপন করার জন্য এ ফাংশন ব্যবহার করা হয়।

১০। গ্রাফিক্স তৈরিতে পিক্সেলের ভূমিকা কী?

উত্তরঃ গ্রাফিক্স তৈরির ক্ষেত্রে কোন ছবি বা গ্রাফিক্সের পিক্সেল সংখ্যা যত বেশি হবে ছবিটি দেখতে তত বেশি সুস্পষ্ট, আকর্ষণীয়, জীবন্ত ও বাস্তব প্রকৃতির হবে। বিপরীতক্রমে পিক্সেল সংখ্যা কম হলে ছবিটি দেখতে ঝাপসা ও অস্পষ্ট প্রকৃতির হবে।

১১। একটি গ্রাফিক্স কার্ডের রেজোলুশন 1024 × 768 এর অর্থ কী?

উত্তরঃ একটি গ্রাফিক্স কার্ডের রেজোলুশন 1024 × 768 এর অর্থ হল গ্রাফিক্স কার্ডটি মনিটরের পর্দায় হরাইজন্টাল বরাবর 1024 টি ও ভার্টিক্যাল বরাবর 768টি পিক্সেল সৃষ্টি করে।

১২। কিভাবে সর্বাধিক রেজোলুশন Video মোড নির্বাচন করা যায়?

উত্তরঃ অ্যাডাপ্টার মোড জানা না থাকলে 'DETECT' ম্যাক্রো ব্যবহার করে সর্বাধিক রেজুলেশন সমৃদ্ধ Video মোড নির্ধারণ করা যায়। এজন্য নিম্নরূপ স্টেটমেন্ট লেখতে হবে।

```
driver = DETECT;
mode = 0;
```

১৩। সরলরেখা তৈরির স্টেটমেন্ট কী? এর ফরম্যাট ও উদাহরণ দাও।

উত্তরঃ সরলরেখা তৈরির জন্য line ( ) ফাংশন ব্যবহৃত হয়।

এর ফরম্যাট হল ঃ

```
line (start x, start y, end x, end y);
```

উদাহরণ ঃ line (50, 50, 150, 150)

১৪। বৃত্ত আঁকার স্টেটমেন্ট কী? এর ফরম্যাট ও উদাহরণ দাও।

উত্তরঃ বৃত্ত আঁকার জন্য circle ( ) ফাংশনটি ব্যবহৃত হয়।

এর ফরম্যাট নিম্নরূপ ঃ

```
circle (x, y, radius);
```

উদাহরণ ঃ circle (400, 200, 100);

১৫। বৃত্তাংশ অঙ্কনের স্টেটমেন্ট লেখ।

উত্তরঃ বৃত্তাংশ অঙ্কনের স্টেটমেন্ট হল arc ( )

এর ফরম্যাট নিম্নরূপ ঃ

```
arc (x, y, start, end, radius);
```

উদাহরণ ঃ arc (200, 150, 0. 90, 50);

১৬। sector ( ) ফাংশনের ব্যবহার লেখ।

উত্তরঃ উপবৃত্তের এরিয়াকে বর্তমানে সক্রিয় রঙ ও ফিল প্যাটার্ন দ্বারা পূর্ণ করার জন্য sector ( ) ফাংশনটি ব্যবহৃত হয়।

এর ফরম্যাট হল ঃ

```
sector (x, y, start, end, x radius, y radius);
```

উদাহরণ ঃ sector (300, 200, 0. 360, 100, 200)

১৭। cleardevice ( ) ফাংশনের কাজ কী?

উত্তরঃ গ্রাফিক্স মোডে আউটপুট স্ক্রীনকে পরিষ্কার করার কাজে cleardevice ( ) ফাংশনটি ব্যবহৃত হয়।

১৮। delay ( ) ফাংশন কখন ব্যবহৃত হয়?

উত্তরঃ প্রোগ্রাম অপারেশনকে কিছু নির্দিষ্ট সময়ের (মিলি সেকেন্ড) জন্য বিরত রাখার জন্য delay ( ) ফাংশনটি ব্যবহৃত হয়।

উদাহরণ ঃ delay (500);

১৯। sound ( ) ফাংশনের কাজ কী?

উত্তরঃ একটি নির্দিষ্ট ফ্রিকুয়েন্সিতে কম্পিউটার স্পিকারে সাউন্ড শুরু করতে sound ( ) ফাংশনটি ব্যবহৃত হয়।
উদাহরণ ঃ sound (100); স্টেটমেন্টের জন্য স্পিকারে 100 হার্টজের একটি সাউন্ড শোনা যাবে।

২০। cprintf ( ) ও cscanf ( ) ফাংশনের কাজ কী?

উত্তরঃ সক্রিয় উইন্ডোতে আউটপুট অপারেশনের জন্য cprintf ( ) এবং ইনপুট অপারেশনের জন্য cscanf () ফাংশন ব্যবহৃত হয়।

২১। মেমোরিতে ইমেজ বা ছবি সংরক্ষণ করার স্টেটমেন্ট ও ফরম্যাট লেখ।

উত্তরঃ মেমোরিতে ইমেজ বা ছবি সংরক্ষণ করার স্টেটমেন্ট হল getimage ( ).
এর ফরম্যাট নিম্নরূপ ঃ

```
getimage (left, top, right, bottom, buf);
```

২২। putimage ( ) ফাংশনের ব্যবহার ও ফরম্যাট লেখ।

উত্তরঃ মেমোরিতে সংরক্ষিত কোন ইমেজ বা ছবিকে পর্দায় লোড করার জন্য putimage ( ) ফাংশনটি ব্যবহৃত হয়।
এর ফরম্যাট হল ঃ

```
putimage (x, y, but, op);
```

২৩। movetext ( ) ফাংশনের কাজ কী?

উত্তরঃ movetext ( ) ফাংশন ব্যবহার করে টেক্সট স্ট্রিংকে স্ক্রীনের এক জায়গা থেকে অন্য জায়গায় স্থানান্তর করা হয়।

➥ *সংক্ষিপ্ত প্রশ্নোত্তর ঃ*

১। গ্রাফিক্স প্রোগ্রামের প্রয়োজনীয়তা কী?
উত্তর সংকেত ঃ ১.০ নং অনুচ্ছেদ দ্রষ্টব্য।

২। গ্রাফিক্স ইনিশিয়ালাইজ করার পদ্ধতি লেখ।
উত্তর সংকেত ঃ ১.০.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। প্রচলিত গ্রাফিক্স ড্রাইভারসমূহের মোড ও রেজোলুশন উল্লেখ কর।
উত্তর সংকেত ঃ ১.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। ellipse ( ) ও fillellipse ( ) ফাংশনের ব্যবহার উল্লেখ কর।
উত্তর সংকেত ঃ ১.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। setcolor ( ) ও setbkcolor ( ) ফাংশনের মধ্যে পার্থক্য লেখ।
উত্তর সংকেত ঃ ১.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। drawpoly ( ) ও fillpoly ( ) ফাংশনের মধ্যে পার্থক্য লেখ।
উত্তর সংকেত ঃ ১.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৭। cprintf ( ) ও cscanf ( ) ফাংশনের ব্যবহার উল্লেখ কর।
উত্তর সংকেত ঃ ১.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৮। সি ভাষায় ইমেজ সংরক্ষণ ও ইমেজ লোড করার জন্য ব্যবহৃত ফাংশনসমূহ ব্যাখ্যা কর।

উত্তর সংকেত ঃ ১.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

৯। টেক্সট স্ট্রিংকে স্থানান্তরের জন্য ব্যবহৃত ফাংশনটি ব্যাখ্যা কর।

উত্তর সংকেত ঃ ১.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

১০। আয়তক্ষেত্র অঙ্কনের জন্য একটি সি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ১.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১১। একটি 3D Bar অঙ্কনের সি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ১.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১২। setfillstyle ( ) ও floodfill ( ) ফাংশনের মধ্যে পার্থক্য কী?

উত্তর সংকেত ঃ ১.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

১৩। একটি বৃত্ত আঁকার সি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ১.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

রচনামূলক প্রশ্নোত্তর ঃ

১। সি ভাষায় মেমোরিতে ইমেজ সংরক্ষণ ও সংরক্ষিত ইমেজকে মেমোরি হতে মনিটর স্ক্রীনে লোড করার পদ্ধতি প্রোগ্রামসহ বর্ণনা কর।

উত্তর সংকেত ঃ ১.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

২। বাংলাদেশের জাতীয় পতাকা অঙ্কনের একটি সি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ১.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। জাপানের জাতীয় পতাকা অঙ্কনের একটি সি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ১.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। টেক্সট স্ট্রিংকে মনিটরের স্থান পরিবর্তন প্রক্রিয়া প্রোগ্রামসহ ব্যাখ্যা কর।

উত্তর সংকেত ঃ ১.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। উদাহরণসহ যে কোন ৫টি গ্রাফিক্স ফাংশন বর্ণনা কর।

উত্তর সংকেত ঃ ১.২ ও ১.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

পলিটেকনিকের সকল বই ডাওনলোড করতে
ভিজিটঃ
www.BDeBooks.Com/polytechnic

সি ভাষায় প্রোগ্রাম ইউটিলিটিজ
Program Utilities in C

অধ্যায় ০২

২.০ ভূমিকা (Introduction) ঃ

C ভাষায় অপারেটিং সিস্টেমের বিভিন্ন ইন্টারাপ্ট সার্ভিস ব্যবহার করে সিস্টেম, হার্ডওয়্যার, ফাইল এবং ডিরেক্টরি সংক্রান্ত বিভিন্ন ধরনের ইউটিলিটিজ প্রোগ্রাম রচনা করা যায়। সিস্টেম সংক্রান্ত ইউটিলিটিজ প্রোগ্রামের ক্ষেত্রে সিস্টেম সংক্রান্ত যাবতীয় ইনফরমেশন জানা, সিস্টেমের ইকুইপমেন্ট লিস্ট বের করা, সিস্টেমের মেমোরি সাইজ নিরূপণ, বর্তমান টাইম নিরূপণ ইত্যাদির কথা উল্লেখ করা যেতে পারে। আবার কী-বোর্ড সংক্রান্ত বিভিন্ন ধরনের প্রোগ্রাম যেমন- কী ডিটেকশন, মাউস সংক্রান্ত নানারকম প্রোগ্রাম যেমন- মাউস পন্টোর তৈরি, প্রিন্টার, ভিডিও (Video) ইউনিট এবং ডিস্ক সংক্রান্ত প্রোগ্রাম, হার্ডওয়্যার এরর ডিটেকশন ইত্যাদি হার্ডওয়্যার সংক্রান্ত ইউটিলিটি প্রোগ্রামের আওতাভুক্ত। তাছাড়াও ডিস্ক এক্সেসিং, বায়োস লেভেল ডিস্ক অপারেশন, ভিডিও মোড নির্ধারণ ইত্যাদি ইউটিলিটি প্রোগ্রামের কথা উল্লেখ করা যায়।

উপরোক্ত সকল ইউটিলিটি প্রোগ্রাম রচনার জন্য C ভাষার ROM BIOS সার্ভিস ব্যবহার করা হয়। ROM BIOS সার্ভিসের জন্য ইন্টারাপ্ট নাম্বার (Interrupt number) প্রয়োজন হয়।

C ভাষায় ROM BIOS প্রোগ্রামের জন্য int86 ( ) ফাংশন ব্যবহার করা হয়।

নিম্নে বিভিন্ন ইন্টারাপ্ট নাম্বারের জন্য ব্যবহৃত ROM BIOS সার্ভিসের একটি তালিকা উল্লেখ করা হল ঃ

ইন্টারাপ্ট নাম্বার (ডেসিমাল)	রম বায়োস সার্ভিস (ডেসিমাল)	[illegible] যন্ত্রাংশ/সার্ভিস
16	0 – F	VDU
17	–	Equipment list
18	–	Memory information
19	0 – 18	Disk services
21	86/88	Device/Memory
22	0 – 2	Keyboard
23	0 – 2	Printer
25	–	Boot strap service
26	0 – 6	Time

২.০.১ প্রোগ্রাম ইউটিলিটিজসমূহ (Program utilities in C) ঃ

সি ভাষায় বহুল ব্যবহৃত প্রোগ্রাম ইউটিলিটিজসমূহ হল ঃ

সিস্টেমের মেমোরি সাইজ নিরূপণ ঃ সিস্টেমের মেমোরি সাইজ নিরূপণের ক্ষেত্রে biosmemory ( ) ফাংশন ব্যবহৃত হয়।

উদাহরণ ঃ

```
int mem-size;
mem-size-biosmemory ( );
```

উপরোক্ত কমান্ডদ্বয়ের মাধ্যমে সিস্টেমের RAM capacity জানা সম্ভব হয়।

সিস্টেমের ইকুইপমেন্ট লিস্ট বের করা ঃ কম্পিউটারে বিদ্যমান ইকুইপমেন্ট লিস্ট নিরূপণের জন্য biosequip ( ) ফাংশন ব্যবহার করা হয়। biosequip ( ) ফাংশনের জন্য <bios. h> হেডার ফাইল include করতে হয়। কম্পিউটারের বিভিন্ন equipment এর তথ্যাবলি ROM এর 2 byte নির্দিষ্ট এলাকায় encoded থাকে।

উদাহরণ ঃ

```
unsigned info;
info = biosequip ( );
```

উপরোক্ত স্টেটমেন্টসমূহের মাধ্যমে কম্পিউটারে বিদ্যমান ইকুইপমেন্টের তথ্য জানা যাবে।

হার্ডওয়্যার জনিত ত্রুটি খুঁজে বের করা ঃ হার্ডওয়্যার সংক্রান্ত কোন সমস্যা হলে অপারেটিং সিস্টেম নানা ধরনের error message দেখায়। যেমন-

'Printer out of paper', 'Drive not ready', 'Data error', 'Invalid disk change', 'Write protect' ইত্যাদি। বেশকিছু লাইব্রেরি ফাংশন ব্যবহার করে এসব হার্ডওয়্যার error খুঁজে বের করা যায়।

```
harderr (handler ( ));
_harderr (handler ( ));
hardresume (code);
_hardresume (code);
hardretn (code);
_hardretn (code); ইত্যাদি।
```

উপরোক্ত সকল ফাংশনসমূহ <dos. h> হেডার ফাইলে declare করা আছে।

ডিস্ক এক্সেস করা ঃ ডিস্ক ড্রাইভ থেকে কোন কিছু রিড করা বা এক্সেস করার জন্য absread ( ) ফাংশন ব্যবহার করা হয়। absread ( ) ফাংশনের মাধ্যমে ডিস্ক এর ফাইল বা ডিরেক্টরি ব্যবহার না করে নির্দিষ্ট কোন সেক্টর হতে তথ্য রিড করা যায়। absread ( ) ফাংশনটি <dos.h> হেডার ফাইলে ডিক্লেয়ার করা আছে।

absread ( ) ফাংশনের ফরম্যাট নিম্নরূপ ঃ

```
absread (drive, numsects, sectnum, buf);
```

এখানে 'drive' হল ডিস্কের যে ড্রাইভ হতে তথ্য read করা হবে তার সংখ্যা মান, 'numsects' হল যে সেক্টর হতে রিড শুরু হবে তার মান এবং 'buf' হল সেই মেমোরি যেখানে রিডকৃত তথ্যসমূহ সংরক্ষিত হবে।

নিম্নে ডিস্ক ড্রাইভের ম্যাক্রো তালিকা উল্লেখ করা হল ঃ

ড্রাইভ	সংখ্যামান
A	0
B	1
C	2
D	3
E	4

টেবিল ঃ ডিস্ক ড্রাইভের ম্যাক্রো তালিকা

বায়োস লেভেল অপারেশনের ক্ষেত্রে ঃ বায়োস লেভেল অপারেশন পরিচালনার জন্য biosdisk ( ) ফাংশন ব্যবহার করা হয়। biosdisk ( ) ফাংশনটি ও ফাইল বা ডিরেক্টরি ব্যবহার না করে সেক্টর নিয়ে কাজ করে। biosdisk ( ) ফাংশনটি নিয়ে কাজ করার জন্য <bios. h> হেডার ফাইল include করতে হয়।

biosdisk ( ) ফাংশনের ফরম্যাট নিম্নরূপ ঃ

```
biosdisk (cmd, drive, head, track, sector, nsects, buf);
```

উদাহরণ ঃ

```
biosdisk (4, 0,.0, 0, 0, 2, buf);
```

উপরোক্ত কমান্ডের মাধ্যমে ড্রাইভ 0 অর্থাৎ ফ্লপি ডিস্কের হেড 0 এর ট্রাক 0 এর সেক্টর 0 থেকে দুইটি সেক্টরের তথ্য রিড করে Buffer এর মধ্যে সংরক্ষণ করবে।

২.১ সফটওয়্যার ইন্টারাপ্ট পদ্ধতি ব্যবহার করে ফাংশন কী ডিটেক্ট করার মূলনীতি (Principle of detecting a function key with software interrupt) ঃ

সফটওয়্যারের মাধ্যমে প্রস্তুতকৃত ইন্টারাপ্ট (Interrupt) সিগন্যালই সফটওয়্যার ইন্টারাপ্ট (Software Interrupt)। অপরদিকে ইন্টারাপ্ট হল এমন এক ধরনের সিগন্যাল যা মাইক্রোপ্রসেসরকে তার স্বাভাবিক কাজ বন্ধ করে অন্য কোন কাজ করতে বাধ্য করে।

C ল্যাংগুয়েজে ইন্টারাপ্ট সিগন্যাল নিয়ে কাজ করার জন্য ROM BIOS সার্ভিস ব্যবহার করা হয়। ROM BIOS সার্ভিস এর মধ্যে এমন কিছু লাইব্রেরী ফাংশন রয়েছে যেগুলো ব্যবহার করে কীবোর্ড সংক্রান্ত বিভিন্ন ধরনের প্রোগ্রাম করা যায়।

কী বোর্ড ও কী বোর্ডের বিভিন্ন কী ডিটেকশন পদ্ধতি ঃ কী বোর্ড ও কী বোর্ডের বিভিন্ন কী ডিটেকশনের জন্য নিম্নবর্ণিত দু'টি লাইব্রেরি ফাংশন ব্যবহার করা হয় যাদের <bios.h> হেডার ফাইলে ডিফাইন করা আছে। যথা ঃ

```
bioskey (cmd);
_ bios_keybrd (cmd);
```

cmd এর মান আবার তিন রকম (0,1,2) হতে পারে। নিম্নে cmd এর ম্যাক্রো ও এর সমতুল্য মান উল্লেখ করা হল।

ম্যাক্রো	সমতুল্য মান
_ KEYBRD _ READ	0
_ KEYBRD _ READY	1
_ KEYBRD _ SHIFTSTATUS	2

টেবিল ঃ

cmd এর মান 0 হলে কীবোর্ড থেকে press করা প্রতিটি কী এর জন্য এর সমতুল্য 16 bit এর একটি ভ্যালু তৈরি হয়। প্রথম 8 bit এ ASCII ক্যারেক্টারের মান থাকে আর পরবর্তী 8 bit এ কী পজিশন এর মান থাকে। cmd এর মান 1 হলে কোন কী press করা হয়েছে কিনা তা পরীক্ষা (Chek) করে। কোন কী Press করা হলে 'non-zero' মান রিটার্ন করে নচেৎ 'zero' মান রিটার্ন করে।

cmd এর মান 2 হলে কী বোর্ড থেকে শিফট কী (Shift key) এর স্ট্যাটাস পাঠানো হয়।

শিফট কী এর স্ট্যাটাস তালিকা নিম্নরূপ ঃ

কী	মান
Right shift	0
Left shift	1
CTRL	2
ALT	3
SCROLL lock on	4
NUM Lock on	5
CAPS Lock on	6
INS on	7

টেবিল ঃ

কী বোর্ডের কী ডিটেক্ট করার প্রোগ্রাম নিম্নরূপ ঃ

```
#include<stdio.h>
#include<conio.h>
#include<bios.h>
#include<ctype.h>
#define RIGHT 0x01
#define LEFT 0x02
#define CTRL 0x04
#define ALT 0x08
int main()
{
int key,modifiers;
clrscr();
while(_bios_keybrd(_KEYBRD_READY)==0);
key=_bios_keybrd(_KEYBRD_READ);
modifiers=_bios_keybrd(_KEYBRD_SHIFTSTATUS);
if(modifiers)
{
printf("[");
if(modifiers & RIGHT)
printf("RIGHT");
if(modifiers & LEFT)
printf("LEFT");
```

```
if(modifiers & CTRL)
printf("CTRL");
if(modifiers & ALT)
printf("ALT");
printf("]");
}
if(isalnum(key & 0xff))
printf("%c\n",key);
else
printf("%#02x\n",key);
getch();
return 0;
}
```

২.২ সি ভাষায় সিস্টেম টাইমারের ব্যবহার (Use of system timer in C)

C ভাষায় সিস্টেমের বর্তমান (Current) সময় জানার জন্য gettime ( ) ফাংশন ব্যবহার করা হয়। এই ফাংশনটি <dos.h> হেডার ফাইলে ডিজাইন করা আছে।

gettime ( ) ফাংশন এর ফরম্যাট নিম্নরূপ ঃ

```
gettime (structime t);
```

এখানে 'time' হল পূর্বে ডিফাইনকৃত একটি স্ট্রাকচার আর 't' হল 'time' নামক স্ট্রাকচারের ভেরিয়েবল।

C ভাষায় পূর্বে ডিফাইনকৃত time নামক স্ট্রাকচারটির ফরম্যাট নিম্নরূপ ঃ

```
struct time
{unsigned char ti-hour;
unsigned char ti-min;
unsigned char ti-sec;
};
```

নিম্নলিখিত প্রোগ্রামের মাধ্যমে সিস্টেমের কারেন্ট টাইম জানা যায়।

```
#include<stdio.h>
#include<conio.h>
#include<dos.h>
void main()
{
struct time t;
clrscr();
while(!kbhit())
{
gettime(&t);
gotoxy(50,30);
printf("Current time is:%2d:%2d:%2d",t.ti_hour,t.ti_min,t.ti_sec);
}
getch();
}
```

২.৩ সি ভাষায় মাউস পয়েন্টার তৈরির পদ্ধতি (Creation of mouse pointer in C)

গ্রাফিকাল ইউজার ইন্টারফেস সিস্টেমে (GUI) মাউস একটি অতি গুরুত্বপূর্ণ ইনপুট ডিভাইস। কারণ মাউস ব্যবহার করে কম্পিউটার সফটওয়্যার বিশেষ করে যেসব সফটওয়্যারে মেনু (Menu) থাকে তাদের নিয়ে কাজ করা অনেক সহজ হয়। তাই C ভাষায় মাউস প্রোগ্রামিং অত্যন্ত জরুরি একটি বিষয়।

মাউস প্রোগ্রামিং এর ক্ষেত্রে বিশেষ বিবেচ্য বিষয়গুলো হচ্ছে ঃ

- রম বায়োস সার্ভিস int86 ( ) ফাংশন এবং ইন্টারাপ্ট নাম্বার Ox33 ব্যবহার করতে হবে।
- মাউস ড্রাইভার লোড করতে হবে। মাউস ড্রাইভার লোড করার ফাংশন হল loaddriver ( )।

একবার মাউস ড্রাইভার লোড হলে পরে বিভিন্ন লাইব্রেরি ফাংশন ব্যবহার করে মাউসের উপর বিভিন্ন প্রোগ্রাম করা যায়।

তাছাড়া ইন্টারাপ্ট নাম্বার Ox33 ব্যবহার করে বিভিন্ন সার্ভিস পাওয়া যায়। নিম্নে Ox33 ইন্টারাপ্ট নাম্বারের সার্ভিসসমূহ উল্লেখ করা হল।

ইন্টারাপ্ট নাম্বার	সার্ভিস নাম্বার	বর্ণনা		
33h	0	Load mouse driver		
33h	1	Show mouse pointer		
33h	2	Hide mouse pointer		
33h	3	Return value on BX register		
		বিট নং	মান	বর্ণনা
		0	1	left button is pressed
		1	1	Right button is pressed
		2	1	Middle button is pressed

টেবিল ঃ

মাউস পয়েন্টার ও মাউস পয়েন্টারের অবস্থান জানার পদ্ধতি ঃ

মাউস পয়েন্টার তৈরি ও মাউস পয়েন্টারের অবস্থান জানার জন্য নিম্নলিখিত ধাপসমূহ অনুসরণ করতে হবে।

- loaddriver ( ) ফাংশন ব্যবহার করে মাউসের জন্য প্রয়োজনীয় ডিভাইস ড্রাইভার লোড করতে হবে।
- এবারে Showmouse ( ) নামে একটি ফাংশন ডিফাইন করা হয়েছে। এর মাধ্যমে মাউস পয়েন্টার তৈরি করা হয়েছে।

```
Showmouse ( )
{
union REGS i, o;
i.x.ax = 1;
int86 (Ox33, & i, & o)
}
```

– মাউস পয়েন্টারের অবস্থান জানার জন্য MousePos ( ) নামের ফাংশন ডিফাইন করা হয়েছে যা নিম্নরূপ ঃ

```
MousePos ( )
{
union REGS i, o;
i. x. ax = 3;
int86 (Ox33; &i, &o);
button = o.x.bx;
x = o.x.cx;
y = o.x.dx;
}
```

Ax রেজিস্টারে সার্ভিস নম্বর 3 নির্ধারণ করলে Bx রেজিস্টারে মাউস বাটনের স্ট্যাটাস পাওয়া যায়। Bx রেজিস্টারের প্রথম বিট এর মান 1 হলে বুঝা যাবে যে, মাউসের left বাটন Press করা হয়েছে। এভাবে প্রতিটি বাটনের স্ট্যাটাস পরীক্ষা করা যায়। মাউস পয়েন্টার তৈরি ও মাউস পয়েন্টারের অবস্থান জানার C প্রোগ্রাম নিম্নরূপ ঃ

```
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<dos.h>
#include<graphics.h>
#define EXITGRAPH   closegraph()
#define NOMOUSE     printf("Mouse Error"); exit(1)
#define LOADGRAPH   initgraph(&driver,&mode,"c:\\tc\\bgi")
int x,y,button;
LoadDriver()
{
union REGS i,o;
i.x.ax=0;
int86(0x33,&i,&o);
return(o.x.ax);
}
ShowMouse()
{
union REGS i,o;
i.x.ax=1;
int86(0x33,&i,&o);
}
MousePos()
{
union REGS i,o;
```

```
i.x.ax=3;
int86(0x33,&i,&o);
button=o.x.bx;
x=o.x.cx;
y=o.x.dx;
}
void main()
{int driver,mode;
driver=DETECT;
LOADGRAPH;
if(LoadDriver()==0)
{
EXITGRAPH;
NOMOUSE;
}
ShowMouse();
printf("Enter any key to exit\n");
while(!kbhit())
{
MousePos();
gotoxy(20,30);
printf("Position of mouse pointer(X,Y) is:%03d,%03d",x,y);
if((button & 1)==1)
{
gotoxy(20,30);
printf("\nButton is pressed");
}
}
}
```

২.৪ গ্রাফিক্স মোডে টেক্সট ও নাম্বার প্রদর্শন পদ্ধতি (Procedure of showing text and numeric values in graphics mode)

গ্রাফিক্স মোডে কোন টেক্সট ও নাম্বার (numeric value) কে বর্তমানে সক্রিয় ফন্ট, ফন্ট সাইজ ও ডিরেকশন অনুযায়ী মনিটরের স্ক্রীনে প্রদর্শনের জন্য outtext( ) ফাংশন ব্যবহৃত হয়। অপরদিকে কোন টেক্সট বা নাম্বারকে মনিটর স্ক্রীনের নির্দিষ্ট কোন লোকেশনে প্রদর্শনের জন্য outtextxy() ফাংশনটি ব্যবহার করা হয়।

outtext ( ) ও outtextxy ( ) ফাংশনের ফরম্যাট নিম্নরূপ ঃ

```
outtext ("Text String");
outtextxy(x, y, String);
```

উপরোক্ত উভয় ফাংশনের ক্ষেত্রে "Text String" বা "String" হল সেই টেক্সট বা নাম্বার যা পর্দায় প্রদর্শন করতে চাই।

আর দ্বিতীয় ফাংশনের ক্ষেত্রে (x, y) হল ডিউপোর্ট বা স্ক্রীনের পজিশন স্থানাঙ্ক যেখানে টেক্সট বা নাম্বারকে ডিসপ্লে করতে চাই।

উদাহরণ ঃ

```
outtext ("Welcome to computer department");
outtextxy (50, 100, welcome to computer department);
```

উপরোক্ত ১ম উদাহরণের জন্য মনিটরের স্ক্রীনে "Welcome to computer department" টেক্সটি প্রদর্শিত হবে। অপরদিকে ২য় উদাহরণের ক্ষেত্রে মনিটর স্ক্রীনের (50, 100) লোকেশনে "Welcome to computer department" টেক্সটি প্রদর্শিত হবে।

নিম্নে একটি প্রোগ্রামের মাধ্যমে গ্রাফিক্স মোডে টেক্সট প্রদর্শন করে দেখানো হল।

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<graphics.h>
void main()
{
int driver,mode;
driver=DETECT;
mode=0;
initgraph(&driver,&mode,"c:\\tc\\bgi");
outtext("Welcome to Computer Department");
outtextxy(50,100,"Welcome to Computer Department");
getch();
}
```

২.৫ সি ভাষায় অবজেক্ট স্থানান্তর পদ্ধতি (Procedure of moving object in C) ঃ

কোন অবজেক্টকে মনিটরের স্ক্রীনের কোন নির্দিষ্ট লোকেশনে স্থানান্তর (Moving) করার জন্য putimage ( ) ফাংশনের ফরম্যাট নিম্নরূপ ঃ

```
putimage (x, y, buf, op);
```

এখানে (x, y) হল সেই স্থানাঙ্ক মান যেখানে অবজেক্ট বা ইমেজটিকে মুভ করতে চাই। 'buf' হল মেমোরি অ্যাড্রেস লোকেশন আর 'op' হল এক ধরনের ম্যাক্রো যার মানের উপর ছবিটির পর্দার রঙ ও ছবিটির রঙ মিলিতভাবে কেমন দেখাবে তা নির্ধারণ করে।

উদাহরণ ঃ

```
putimage (10, 10, buf, copy-put);
```

উপরোক্ত স্টেটমেন্টের মাধ্যমে 'buf' নামক মেমোরি অ্যাড্রেসে সংরক্ষিত কোন অবজেক্টকে মনিটর স্ক্রীনের (10, 10) স্থানাঙ্ক পজিশনে কপিকৃত ছবিটি হুবহু প্রদর্শিত হবে।

নিম্নে একটি প্রোগ্রামের মাধ্যমে কোন একটি অবজেক্টকে স্থানান্তর করে দেখানো হল।

একটি অবজেক্ট (আয়তক্ষেত্র) অংকণ করে উহাকে অ্যারো কী দ্বারা মুভ করার প্রোগ্রাম ঃ

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#define ignored 0
int midx,midy,right,bottom;
void rec(int left, int top, int right, int bottom, int color);
void main()
{
int driver,mode;
driver=DETECT;
mode=0;
int left,top,color,ch;
initgraph(&driver,&mode,"c:\\tc\\bgi");
printf("Press Esc to exit");
midx=getmaxx()/2;
midy=getmaxy()/2;
left=midx+100; top=midy+100;
right=midx-100; bottom=midy-100;
setlinestyle(SOLID_LINE,ignored,THICK_WIDTH);
rec(left,top,right,bottom,3);
do
{
ch=getch();
if(ch==80)
{
rec(left,top,right,bottom,0);
left=left;top=top+1;right=right;bottom=bottom+1;
rec(left,top,right,bottom,3);
}
if(ch==72)
{
rec(left,top,right,bottom,0);
left=left;top=top-1;right=right;bottom=bottom-1;
rec(left,top,right,bottom,3);
```

```
}
if(ch==75)
{
rec(left,top,right,bottom,0);
left=left-1;top=top;right=right-1;bottom=bottom;
rec(left,top,right,bottom,3);
}
if(ch==77)
{
rec(left,top,right,bottom,0);
left=left+1;top=top;right=right+1;bottom=bottom;
rec(left,top,right,bottom,3);
}
}while(ch!=27);
getch();
}
void rec(int left, int top, int right, int bottom, int color)
{
setcolor(color);
rectangle(left,top,right,bottom);
}
```

২.৬ সি ভাষায় ত্রিমাত্রিক অবজেক্ট তৈরির পদ্ধতি (Procedure of creating 3D object in C) ঃ

সি ভাষায় ত্রিমাত্রিক অবজেক্ট তৈরির জন্য বিভিন্ন ধরনের গ্রাফিক্স ফাংশনের ব্যবহার জানতে হবে যা ১ম ও ২য় অধ্যায়ে ধারাবাহিকভাবে আলোচনা করা হয়েছে। নিম্নে কয়েকটি ত্রিমাত্রিক অবজেক্ট তৈরির প্রোগ্রাম লেখা হল ঃ

ত্রিমাত্রিক অবজেক্ট (3D Object) তৈরির সি প্রোগ্রাম

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<dos.h>
#include<graphics.h>
#include<iostream.h>
void main()
{
```

```
int driver,mode;
int ix,iy,r1x1,r1x2,r1x3,r1x4,r1y1,r1y2,r1y3,r1y4;
int r2x1,r2x2,r2x3,r2x4,r2y1,r2y2,r2y3,r2y4;
int poly[8];
driver=DETECT;
mode=0;
initgraph(&driver,&mode,"c:\\tc\\bgi");
ix=20;iy=20;r1x1=100;r1y1=100;r1x2=100;
r1y2=200;r1x3=20;r1y3=200;r1x4=200,r1y4=100;
r2x1=r1x1+ix; r2y1=r1y1+iy;
r2x2=r1x2+ix; r2y2=r1y2+iy;
r2x3=r1x3+ix; r2y3=r1y3+iy;
r2x4=r1x4+ix; r2y4=r1y4+iy;
poly[0]=r1x1; poly[1]=r1y1;
poly[2]=r2x1; poly[3]=r2y1;
poly[4]=r2x4; poly[5]=r2y4;
poly[6]=r1x4; poly[7]=r1y4;
setcolor(2);
setfillstyle(1,2);
fillpoly(4,poly);drawpoly(4,poly);
poly[0]=r1x1; poly[1]=r1y1;
poly[2]=r1x2; poly[3]=r1y2;
poly[4]=r2x2; poly[5]=r2y2;
poly[6]=r2x1; poly[7]=r2y1;
setcolor(4);
setfillstyle(1,4);
fillpoly(4,poly);drawpoly(4,poly);
poly[0]=r2x1; poly[1]=r2y1;
poly[2]=r2x2; poly[3]=r2y2;
poly[4]=r2x3; poly[5]=r2y3;
poly[6]=r2x4; poly[7]=r2y4;
setcolor(14);
setfillstyle(1,14);
fillpoly(4,poly);drawpoly(4,poly);
getch();
}
```

একটি মুভিং আয়তক্ষেত্রের মধ্যে **"Programming Language-2"** বাক্যটি প্রদর্শন করবে এমন একটি স্ক্রীন সেভার তৈরির প্রোগ্রাম ঃ

```
#include<stdio.h>
#include<conio.h>
#include<dos.h>
#include<graphics.h>
void main()
{
int i;
int driver,mode;
driver=DETECT;
initgraph(&driver,&mode,"c:\\tc\\bgi");
i=1;
do
{
outtextxy(280+i,80+i,"Programming Language-2");
setcolor(i);
rectangle(270+i,40+i,470+i,140+i);
delay(100);
cleardevice();
i++;
if(i==100)
i=1;
}while(!kbhit());
getch();
closegraph();
}
```

একটি আয়তাকার ক্ষেত্রের মধ্যে একটি মুভিং বল (moving ball) প্রদর্শন করার সি প্রোগ্রাম ঃ

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#define left 0
#define top 0
#define right 400
#define bottom 349
#define radius 20
void main()
{
int driver,mode;
int x,y,dx,dy,oldx,oldy;
unsigned char ballbuf[2000];
driver=DETECT;
mode=0;
initgraph(&driver,&mode,"c:\\tc\\bgi");
rectangle(left,top,right,bottom);
x=left+radius+10;
y=top+radius+10;
setcolor(LIGHTGREEN);
setfillstyle(SOLID_FILL, LIGHTGREEN);
circle(x,y,radius);
floodfill(x,y,LIGHTGREEN);
getimage(x-radius,y-radius,x+radius,y+radius,ballbuf);
dx=2; dy=2;
while(!kbhit())
{
putimage(x-radius,y-radius,ballbuf,COPY_PUT);
oldx=x; oldy=y;
x=x+dx; y=y+dy;
if(x<=left+radius+2||x>=right-radius-2)
dx=-dx;
if(y<=top+radius+2||y>=bottom-radius-2)
dy=-dy;
putimage(oldx-radius,oldy-radius,ballbuf,XOR_PUT);
}
getch();
}
```

অনুশীলনী-২

➥ *অতি সংক্ষিপ্ত প্রশ্নোত্তর ঃ*

১। **ROM BIOS প্রোগ্রামিং কী?**

উত্তরঃ সি ভাষায় রম বায়োস সার্ভিস ব্যবহার করে হার্ডওয়্যার সংক্রান্ত বিভিন্ন ধরনের ইউটিলিটিজ প্রোগ্রাম রচনার পদ্ধতিই হচ্ছে ROM BIOS প্রোগ্রামিং।

২। **ইন্টারাপ্ট কী?**

উত্তরঃ ইন্টারাপ্ট হল এমন এক ধরনের সিগনাল যা মাইক্রোপ্রসেসরকে তার স্বাভাবিক কাজকর্ম বন্ধ রেখে অন্য কোন কাজ করতে বাধ্য করে।

৩। **সফটওয়্যার ইন্টারাপ্ট বলতে কী বোঝায়?**

উত্তরঃ সফটওয়্যারের মাধ্যমে প্রস্তুতকৃত ইন্টারাপ্ট সিগনালকে সফটওয়্যার ইন্টারাপ্ট বলে।

৪। **ROM BIOS সার্ভিস কী?**

উত্তরঃ রম বায়োস সার্ভিস হল এমন কতগুলো লাইব্রেরি ফাংশনের সংগ্রহ যাদের মাধ্যমে হার্ডওয়্যার সংক্রান্ত বিভিন্ন ধরনের প্রোগ্রাম করা যায়।

৫। **মাউস প্রোগ্রামিং বলতে কী বোঝায়?**

উত্তরঃ বিভিন্ন ধরনের লাইব্রেরি ফাংশন ব্যবহার করে মাউস সংক্রান্ত বিভিন্ন ধরনের অপারেশন (যেমন ঃ মাউস পয়েন্টার তৈরি, মাউস পয়েন্টারের অবস্থান নির্ণয়) সম্পন্ন করার জন্য প্রোগ্রাম রচনা করার পদ্ধতিকে মাউস প্রেগ্রামিং বলে।

৬। **মাউস প্রোগ্রামিং এর ক্ষেত্রে বিবেচ্য বিষয়সমূহ কী কী?**

উত্তরঃ মাউস প্রোগ্রামিং এর ক্ষেত্রে বিবেচ্য বিষয়গুলো হচ্ছে–

- রম বায়োস সার্ভিস int 86 ( ) ফাংশন ও ইন্টারাপ্ট নাম্বার 0×33 ব্যবহার করতে হবে।
- মাউস ড্রাইভার লোড করতে হবে।

৭। **সিস্টেম টাইমার এর কাজ কী?**

উত্তরঃ সিস্টেমের বর্তমান সময় জানার জন্য সিস্টেম টাইমার ব্যবহৃত হয়। সিস্টেমের বর্তমান সময় জানার জন্য gettime ( ) ফাংশনটি ব্যবহৃত হয়।

৮। **outtext ( ) ও outtextxy ( ) ফাংশনের মধ্যে পার্থক্য কী?**

উত্তরঃ গ্রাফিক্স মোডে কোন টেক্সট ও নিউমেরিক ভ্যালুকে বর্তমানে সক্রিয় ফন্ট, ফন্ট সাইজ ও ডিরেকশন অনুযায়ী মনিটর স্ক্রীনে প্রদর্শনের জন্য outtext ( ) ফাংশন ব্যবহৃত হয়। অপরদিকে কোন টেক্সট ও নিউমেরিক ভ্যালুকে মনিটরের নির্দিষ্ট কোন লোকেশনে প্রদর্শনের জন্য outtextxy ( ) ফাংশনটি ব্যবহৃত হয়।

৯। **কোন অবজেক্টকে কিভাবে মনিটরের নির্দিষ্ট লোকেশনে স্থানান্তর করা হয়?**

উত্তরঃ putimage ( ) ফাংশন ব্যবহার করে কোন অবজেক্টকে মনিটরের নির্দিষ্ট কোন লোকেশনে স্থানান্তর করা হয়।

১০। **সিস্টেমের মেমোরি সাইজ নিরুপণের জন্য ব্যবহৃত ফাংশনের নাম লেখ।**

উত্তরঃ সিস্টেমের মেমোরি সাইজ নিরুপনের জন্য biosmemory ( ) ফাংশনটি ব্যবহৃত হয়।

১১। **biosequip ( ) ফাংশনের কাজ কী?**

উত্তরঃ biosequip ( ) ফাংশনের মাধ্যমে সিস্টেমে ব্যবহৃত ইকুইপমেন্ট লিস্ট বের করা যায়।

১২। কয়েকটি লাইব্রেরি ফাংশনের নাম লেখ যাদের ব্যবহার করে হার্ডওয়্যার সংক্রান্ত ত্রুটি খুঁজে বের করা যায়?

উত্তরঃ হার্ডওয়্যার সংক্রান্ত ত্রুটি খুঁজে বের করার লাইব্রেরি ফাংশনসমূহ হচ্ছে–

```
harderr (handler ( ) );
_harderr (handler ( ) );
hardresume (code);
_hardresume (code);
hardretn (code);
_hardretn (code); ইত্যাদি।
```

সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। সি ভাষায় বহুল ব্যবহৃত প্রোগ্রাম ইউটিলিটিজসমূহ কী কী?

উত্তর সংকেত ঃ ২.০.১ নং অনুচ্ছেদ দ্রষ্টব্য।

২। ইন্টারাপ্ট নাম্বার 33h এর সার্ভিসগুলো কী কী?

উত্তর সংকেত ঃ ২.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। কীবোর্ড ও কীবোর্ডের কী ডিটেকশন পদ্ধতি উল্লেখ কর।

উত্তর সংকেত ঃ ২.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। Time স্ট্রাকচারের ফরম্যাট উল্লেখ কর।

উত্তর সংকেত ঃ ২.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। মাউস পয়েন্টার তৈরি ও মাউস পয়েন্টারের অবস্থান নির্ণয় করা যায় কিভাবে?

উত্তর সংকেত ঃ ২.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

রচনামূলক প্রশ্নোত্তর ঃ

১। সফটওয়্যার ইন্টারাপ্ট পদ্ধতি ব্যবহার করে কী বোর্ডের কী ডিটেক্ট করার সি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ২.১ নং অনুচ্ছেদ দ্রষ্টব্য।

২। মাউস পয়েন্টার তৈরি করার জন্য একটি সি ভাষার প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ২.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। মনিটরের নির্দিষ্ট কোন লোকেশনে অবজেক্ট স্থানান্তরের প্রক্রিয়াটি একটি প্রোগ্রাম লিখে ব্যাখ্যা কর।

উত্তর সংকেত ঃ ২.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। একটি 3D অবজেক্ট তৈরির সি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ২.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। Screen saver তৈরির একটি সি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ২.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। উইন্ডোর মধ্যে অবজেক্ট নড়াচড়ার জন্য একটি সি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ২.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

অধ্যায় ০৩

অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং
Object Oriented Programming

৩.০ ভূমিকা (Introduction) ঃ

কম্পিউটার আবিষ্কারের পর থেকেই তা দিয়ে বিভিন্নভাবে জটিল সমস্যা সহজে সমাধানের চেষ্টা অব্যাহত রয়েছে এবং গবেষণা ও উন্নয়নমূলক কাজ চলছে বিভিন্ন প্রোগ্রামিং পদ্ধতি নিয়ে। মডুলার প্রোগ্রামিং, টপ-ডাউন প্রোগ্রামিং, বটম-আপ প্রোগ্রামিং উল্লেখযোগ্য কয়েকটি প্রোগ্রামিং পদ্ধতি। সকল প্রোগ্রামিং পদ্ধতির উদ্দেশ্য অভিন্ন– সহজে সমস্যা সমাধানের জন্য প্রোগ্রাম তৈরি, প্রোগ্রামের গ্রহণযোগ্যতা বৃদ্ধি, ব্যবহৃত ডাটার সহজ সংরক্ষণ ও গোপনীয়তা রক্ষা ইত্যাদি। আশির দশকে সি ভাষার আবির্ভাবের পর প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং ব্যাপক জনপ্রিয়তা পায়। তবে সমস্যার পরিধি বৃদ্ধির সাথে সাথে প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং পদ্ধতির ধারণাসহ অতিরিক্ত কিছু ধারণা নিয়ে প্রোগ্রামিং জগতে নতুন চমক নিয়ে আসে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং পদ্ধতি।

বাস্তব ভিত্তিক (Real time) সমস্যা সমাধানের লক্ষ্যেই অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর আবির্ভাব। Simula67ই সর্বপ্রথম অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ, ১৯৬০ সালে যার প্রচলন শুরু হয়। পরবর্তীতে আসে VHDL। তারও অনেক পরে ১৯৮০ সালে যুক্তরাষ্ট্রের এটিএন্ডটি বেল ল্যাবরেটরি (AT&T Bell laboratory) ও জর্ন স্ট্রাউস্ট্রপ (Bjarne stroustrop) এর যৌথ উদ্যোগে পরিচালিত simula 67 এবং C এর সমন্বয়ে গঠিত গবেষণার ফলে নতুন এক ল্যাংগুয়েজের উদ্ভব ঘটে যার নাম C++। এটি C এর মত প্রসিডিওর প্রোগ্রামিং এর সুবিধা দেয় এবং অবজেক্ট অরিয়েন্টেড পদ্ধতির ব্যবহার ও অনুমোদন করে।

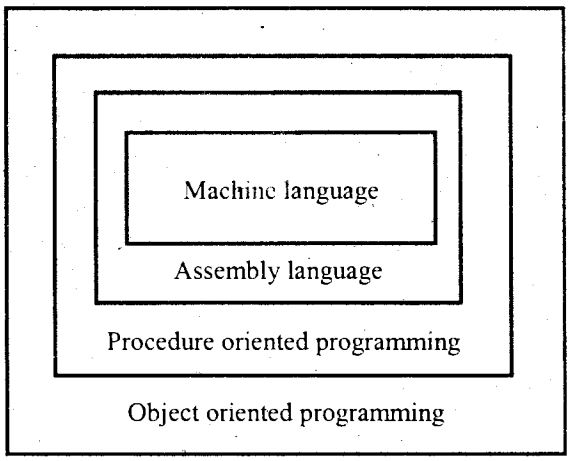
প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং এবং অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং পদ্ধতির মধ্যে ব্যাপক পার্থক্য রয়েছে। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং-এর তুলনায় সাধারণ প্রসিডিউর প্রোগ্রামিং সম্পূর্ণ অচল। প্রসিডিউর প্রোগ্রাম ভাষায় যেখানে ফাংশন বা প্রসিডিউর লিখে প্রোগ্রাম তৈরি করা হয়; অবজেক্ট অরিয়েন্টেড প্রোগ্রাম ভাষায় সেখানে ক্লাস এবং অবজেক্ট ডিজাইন করে প্রোগ্রাম তৈরি করা হয়। একবার কোন ক্লাস ডিজাইন করে তা ক্লাস লাইব্রেরিতে রেখে সেই ক্লাস পুণঃব্যবহার করা যায়। প্রয়োজনে পুরাতন ক্লাস পরিবর্ধন করে অপর কোন প্রোগ্রামে ব্যবহার করা যায়। এতে মূল ক্লাসের সকল বৈশিষ্ট্য অক্ষুণ্ন থাকে, কিন্তু নতুন ক্লাসের বৈশিষ্ট্য বৃদ্ধি পায়। আবার অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং-এর ভিজুয়াল (দৃশ্যমান) সংস্করণগুলোতে (যেমন– ভিজুয়াল ফক্সপ্রো, ভিজুয়াল বেসিক, ভিজুয়াল সি++ ইত্যাদি) মাউস ক্লিক অথবা মাউস ড্রাগ করে কোন অবজেক্ট বা ক্লাস অতি সহজেই অপর কোন প্রোগ্রামে ব্যবহার করা যায়। ফলে প্রোগ্রামের আকার ছোট হয় এবং গ্রহণযোগ্যতা বৃদ্ধি পায়।

৩.১ সফটওয়্যার বিবর্তন (Software Evolution) ঃ

সৃষ্টিজগতের প্রতিটি ক্ষেত্রে বিবর্তন (Evolution) একটি অত্যাবশ্যকীয় শব্দ। বর্তমানের চাহিদা মেটাতে পরিবর্তন বা বিবর্তনের মাধ্যমে উন্নয়নের কোন বিকল্প নেই। তবে যে জগতে এ পরিবর্তনের মাত্রা সর্বাধিক এবং নৈমিত্তিক সেটি হচ্ছে সফটওয়্যার জগৎ।

প্রখ্যাত আর্টিফিসিয়াল ইন্টেলিজেন্স (Artificial Intelligence) প্রযুক্তির লেখক আর্নেস্ট টেলোর (Earnest tello) মতে সফটওয়্যার জগতের বিবর্তন একটি গাছের বেড়ে ওঠার মতই। একটি গাছ যেমন কয়েকটি ধাপে পূর্ণাঙ্গরূপে পরিণত হয় তেমনি আজকের এই আধুনিক ও সুগোপযোগী সফটওয়্যার টেকনোলজিও পূর্বের কয়েকটি ধাপের বিবর্তনেরই ফসল। মূলত আজকের সফটওয়্যার টেকনোলজির বিবর্তন 4টি ধাপে সম্পন্ন হয়েছে। ধাপগুলো হচ্ছে ঃ

- মেশিন ল্যাংগুয়েজ (Machine language)
- এসেম্বলি ল্যাংগুয়েজ (Assembly language)
- প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং (Procedure oriented programming)
- অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং (Object oriented programing).



চিত্র ঃ সফটওয়্যার টেকনোলজির বিবর্তন

মেশিন ল্যাংগুয়েজ (Machine language) ঃ যে ভাষায় সমস্যা সমাধানের জন্য বাইনারি সংখ্যা (0,1) ব্যবহার করে নির্দেশ (Instruction) সাজিয়ে প্রোগ্রাম রচনা করা হয়, তাই মেশিন ল্যাংগুয়েজ (Machine language)।

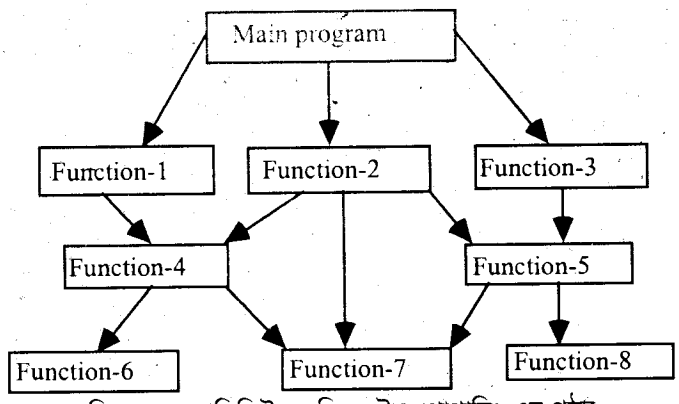
- মেশিন ল্যাংগুয়েজে লিখিত প্রোগ্রামকে অবজেক্ট প্রোগ্রাম (Object program) বলে।
- এ ভাষায় প্রোগ্রাম লিখা তুলনামূলকভাবে কষ্টসাধ্য এবং সময়সাপেক্ষ। এ ভাষায় লিখিত প্রোগ্রাম নির্বাহের জন্য কোন অনুবাদক প্রোগ্রামের (Translator) প্রয়োজন হয় না।
- এক ধরনের মেশিনের জন্য লিখিত প্রোগ্রাম অন্য ধরনের মেশিনে ব্যবহার করা যায় না।
- এ ভাষায় প্রোগ্রাম লেখার জন্য কম্পিউটারে প্রতিটি ইন্সট্রাকশন এবং মেমোরি অ্যাড্রেস সম্বন্ধে পরিষ্কার ধারণা থাকা অপরিহার্য।

এসেম্বলি ল্যাংগুয়েজ (Assembly language) ঃ বিশেষ সংক্ষিপ্ত সংকেত বা নেমোনিক (ADD, SUB, MUL, DIV, LOAD, MOV, MVI, JMP, JNZ, INC ইত্যাদি) ব্যবহার করে যে ভাষায় প্রোগ্রাম রচনা করা হয় তাকে এসেম্বলি ল্যাংগুয়েজ (Assembly language) বলে। যেমন- MASM, TASM ইত্যাদি।

- এ ভাষায় লিখিত প্রোগ্রামকে সোর্স প্রোগ্রাম (Source program) বলে।
- মেশিন ল্যাংগুয়েজের তুলনায় এসেম্বলি ল্যাংগুয়েজে প্রোগ্রাম রচনা অপেক্ষাকৃত সহজ ও কম সময়সাপেক্ষ হলেও হাই লেভেল ল্যাংগুয়েজের তুলনায় কষ্টসাধ্য এবং সময় সাপেক্ষ।
- এ ভাষায় প্রোগ্রাম রচনার জন্য প্রচুর কমান্ড (Command) মুখস্থ রাখতে হয়।
- এ ভাষায় লিখিত প্রোগ্রাম নির্বাহের জন্য অনুবাদক প্রোগ্রামের (Assembler) প্রয়োজন হয়।

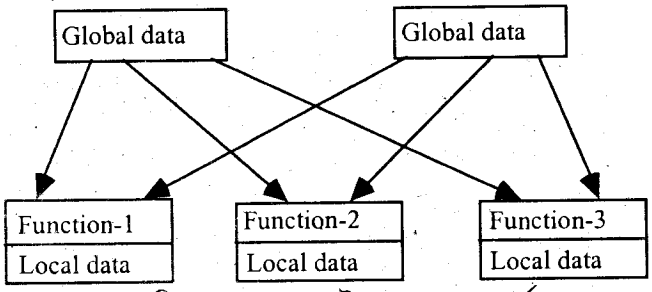
প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং (Procedure oriented programming) ঃ

মূলতঃ হাই লেভেল ল্যাংগুয়েজ যেমন– COBOL, FORTRAN, BASIC, C ইত্যাদি ব্যবহার করে যে প্রোগ্রাম তৈরি করা হয় তাকে স্ট্রাকচার্ড প্রোগ্রামিং কিংবা প্রসিডিউর অরিয়েন্টেড প্রোগ্রাম (POP) বলে। এ প্রোগ্রামের প্রধান বিষয়বস্তু হচ্ছে- ফাংশন। এখানে ফাংশন ডাটা নিয়ে কাজ করে এবং ডাটার চেয়ে ফাংশনের গুরুত্ব বেশি থাকে। পঞ্চাশের দশকের দিকে যখন Assembly, FORTRAN প্রভৃতি ল্যাংগুয়েজ বের হয় তখন হাজার হাজার লাইনের প্রোগ্রাম খুব সহজে লেখা যেত। কিন্তু এই সকল language ব্যবহার করে লিখিত প্রোগ্রামগুলি কোডিং করা ছিল খুবই অসুবিধাজনক। এই সকল অসুবিধা দূর করার জন্যই ষাটের দশকে Structured বা Procedure Oriented Programming language তৈরি করা হয়। যেমনঃ Algol, Pascal ইত্যাদি। এই প্রোগ্রামিং পদ্ধতিতে একটি বড় প্রোগ্রামকে বা সমস্যাকে কতগুলো ছোট ছোট অংশ বা মডিউলে ভাগ করা হয়। সমস্যা সমাধানের উপর ভিত্তি করে প্রতিটি অংশের জন্য আলাদা আলাদা ফাংশন ব্যবহার করা হয় এবং পরবর্তীতে সকল ফাংশন সমূহকে একত্রিত করে সমস্যার সমাধান করা হয়।



চিত্র ঃ ১.২ প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং এর গঠন

সাধারণতঃ ফাংশনসমূহকে সাজানো কিংবা উহাদের একে অপরের মধ্যকার সম্পর্ক একটি Flow chart এর মাধ্যমে represent করা যায় (চিত্র ঃ ১.৩)।



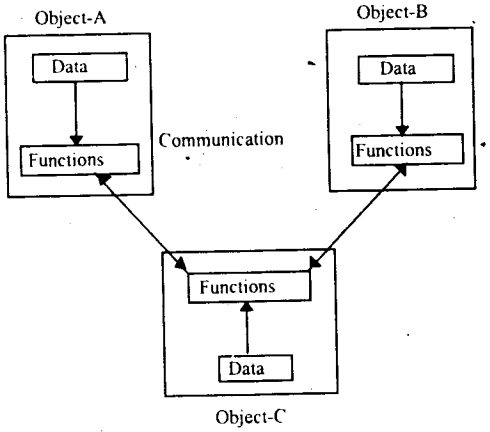
চিত্র ঃ ফাংশন ও ডাটা'র মধ্যকার সম্পর্ক

উপরের চিত্র হতে বুঝা যায় যে, Multi function প্রোগ্রামিং এর ক্ষেত্রে প্রায় প্রতিটি ফাংশনের জন্য Global data ব্যবহার করা হয় যাতে একই ডাটা বিভিন্ন ফাংশন ব্যবহার করতে পারে। ফলে এতে System এর Security হ্রাস পায়। বড় বড় প্রোগ্রামের ক্ষেত্রে কোন ফাংশনের জন্য কোন ডাটা ব্যবহৃত হচ্ছে তা সনাক্ত করা বেশ কষ্টকর। আবার যদি কখনো বড় কোন প্রোগ্রাম স্ট্রাকচার থেকে নির্দিষ্ট কোন ডাটা খুঁজে বের করতে চাই তাহলে পুরো প্রোগ্রাম স্ট্রাকচারকে পরীক্ষা করে দেখতে হয়। তাই এ প্রক্রিয়াটি বেশ সময় সাপেক্ষ ও বিরক্তিকর। এর অন্যতম প্রধান একটি সমস্যা হচ্ছে, এতে ব্যবহৃত ফাংশন সমূহ Action oriented যা বাস্তব জীবনের সমস্যা সমাধানের সাথে খুব বেশি সম্পর্কিত নয়। এধরনের ল্যাংগুয়েজে প্রোগ্রামের আকার যখন বেড়ে যায় তখন এর রক্ষণাবেক্ষন, ভুল-সংশোধন, পরিবর্তন, পরিবর্ধন ও পরিমার্জন ইত্যাদি অনেক কঠিন হয়ে পড়ে।

- এ ভাষায় একটি বড় প্রোগ্রামকে কতগুলো ক্ষুদ্র ক্ষুদ্র অংশে বিভক্ত করা হয়, প্রতিটি ক্ষুদ্র অংশ 'ফাংশন' নামে পরিচিত।
- প্রত্যেকটি ফাংশনই গ্লোবাল ডাটা শেয়ার করে।
- ডাটা সহজেই এক ফাংশন থেকে অন্য ফাংশনে Move করতে পারে।
- এক ফাংশনের ডাটা অন্য ফাংশনে সহজেই স্থানান্তরিত হতে পারে।
- এই ধরনের প্রোগ্রাম 'Top down' পদ্ধতিতে Design করা হয়।
- প্রত্যেকটি ফাংশনের জন্য যেমন গ্লোবাল ডাটা রয়েছে, পাশাপাশি লোকাল ডাটা ও রয়েছে।

অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং (Object oriented programming) ঃ প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজে ডাটার চেয়ে ফাংশনের গুরুত্ব বেশি এবং যখনই এ ধরনের প্রোগ্রামের আকার বেড়ে যায় তখন এর রক্ষণাবেক্ষণ, ভুল সংশোধন, পরিমার্জন, পরিবর্তন, পরিবর্ধন ইত্যাদি অনেকটা কঠিন হয়ে পড়ে। প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং এ বিদ্যমান এসব সমস্যাবলি দূর করার জন্য যে প্রোগ্রামিং এর আবির্ভাব তাই অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং (OOP)।

অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এ একটি বৃহৎ সমস্যা (Problem) কে ছোট ছোট অংশে বিভক্ত করা হয়। ছোট ছোট অংশগুলিকে Sub problem বা Object বলা হয়। OOP-তে ফাংশনের চেয়ে ডাটার ওপর বেশি গুরুত্বারোপ করা হয়। প্রতিটি Object এর জন্যই কিছু Data থাকে এবং ঐ ডাটাগুলোকে কাজ করানোর জন্য specific কিছু ফাংশন ও থাকে। উল্লেখ্য যে, কোন অবজেক্টের ফাংশন অন্য কোন অবজেক্টের ফাংশনকে নিয়েও কাজ করতে পারে। নিম্নে একটি OOP Organization এ data ও function এর মধ্যকার সম্পর্ক চিত্র তুলে ধরা হলো।



চিত্র ঃ OOP-তে data ও function এর মধ্যকার সম্পর্ক

- এই ধরনের system এ function এর চেয়ে data'র গুরুত্ব বেশি।
- একটি বড় প্রোগ্রামকে কতগুলো ছোট ছোট অংশে বিভক্ত করা হয়। প্রতিটি ক্ষুদ্র ক্ষুদ্র অংশ object হিসাবে পরিচিত।
- সমস্যার ধরণ অনুযায়ী data structure এর জন্য নিজস্ব data ও function ব্যবহার করা হয়।
- এতে সবগুলি ক্ষুদ্র ক্ষুদ্র অংশসমূহকে একত্রিত করে সমস্যার সমাধান করা হয়।
- একটি object এর data কে অন্য কোন object এর function দ্বারা access করা যায় না।
- এতে অবজেক্টসমূহ ফাংশন ব্যবহার করে একে অপরের সাথে যোগাযোগ রক্ষা করতে পারে।
- যে কোন সময়ে প্রয়োজন অনুযায়ী data কিংবা function, structure এ add করা যাবে।
- এতে বটম আপ পদ্ধতিতে program design করা হয়।
- এতে Message passing এর মাধ্যমে এক অবজেক্ট অন্য অবজেক্টের সাথে সংযোগ রক্ষা করে।
- এছাড়াও object oriented programming এ Class, Object, Inheritance, Data Abstraction, Data Hiding, Encapsulation, Polymorphism, Static Binding, Message passing, Concurrency, Object Persistance, Object Library ইত্যাদি বৈশিষ্ট্য বিদ্যমান।

৩.১.১ প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং ও অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর মধ্যে পার্থক্য (Difference between procedure oriented programing and object oriented programming) ঃ

নিম্নে প্রসিডিউর অরিয়েন্টেড ও অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর মধ্যকার পার্থক্য তুলে ধরা হল ঃ

প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং	অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং
মূলতঃ High level language যেমন ঃ Cobol, Fortran, Basic, C ইত্যাদি ভাষা ব্যবহার করে যে Program রচিত তাকে প্রসিডিওর অরিয়েন্টেড প্রোগ্রামিং বলে।	প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং এ বিদ্যমান অসুবিধা সমূহ সমাধানের নিমিত্তে নতুন ধরনের যে প্রোগ্রামিং এর আবির্ভাব তাকে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং বলে।
মূল প্রোগ্রামকে কতগুলো module এ ভাগ করা হয়। পরবর্তীতে module গুলোর function সমূহকে একত্রিত করে সমস্যার সমাধান করা হয়।	মূল প্রোগ্রামকে এখানে ও কতগুলো ছোট ছোট অংশে মানে object এ ভাগ করা হয়। পরবর্তীতে object সমূহকে একত্রিত করে সমস্যার সমাধান করা হয়।
এতে ডাটা অপেক্ষা ফাংশনের গুরুত্ব বেশি।	এতে ফাংশন অপেক্ষা ডাটার গুরুত্ব বেশি।
এই ধরনের প্রোগ্রামিং টপ ডাউন পদ্ধতিতে করা হয়।	এই ধরনের প্রোগ্রামিং বটম আপ পদ্ধতিতে করা হয়।
সাধারণ নিয়মে এক ফাংশন অন্য ফাংশন-এর সাথে সংযোগ রক্ষা করে।	মেসেজ পাসিং এর মাধ্যমে এক অবজেক্ট অন্য অবজেক্টের সাথে সংযোগ রক্ষা করে।
এতে প্রতিটি ফাংশনের জন্য আলাদা আলাদা ডাটা থাকে যাকে লোকাল ডাটা বলে।	এতে প্রতিটি অবজেক্টের জন্য ডাটা ও ফাংশনের সমন্বয়ে একটি একক ইউনিট থাকে।

প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং	অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং
ডাটা সহজেই এক ফাংশন থেকে অন্য ফাংশনে মুভ করতে পারে।	এতে সহজেই এক অবজেক্টের ডাটা অন্য অবজেক্টে মুভ করতে পারে না।
এই ধরনের প্রোগ্রামিং এ ভুল-ত্রুটি নির্ণয় ও এদের সমাধান বেশ কঠিন।	এই ধরনের প্রোগ্রামিং এ ভুল-ত্রুটি নির্ণয় ও এদের সমাধান অপেক্ষাকৃত অনেক সহজ।
বেশির ভাগ ফাংশন গ্লোবাল ডাটা শেয়ার করে বলে সিস্টেমের সিকিউরিটি হ্রাস পায়।	এই ধরনের প্রোগ্রামিং এ সিস্টেমের সিকিউরিটি অনেক বেশি।
এই পদ্ধতিতে বড় বড় প্রোগ্রাম লিখতে গিয়ে data abstruction, code reusability ইত্যাদি বিষয়ে সমস্যায় পড়তে হয়।	এই পদ্ধতিতে এই ধরনের সমস্যায় পড়ার সম্ভাবনা কম।
এই পদ্ধতিতে প্রোগ্রাম এর মেইনটেন্যান্স করা অনেক কঠিন।	এই পদ্ধতিতে প্রোগ্রাম মেইনটেন্যান্স অনেক সহজ।
এতে ব্যবহৃত ফাংশনসমূহ Action oriented যা বাস্তব জীবনের সাথে সম্পর্কিত নয়।	এতে ব্যবহৃত অবজেক্টসমূহ object oriented যা বাস্তব জীবনের সাথে সম্পর্কিত।
উদাহরণ ঃ ফরট্রান প্যাস্কেল, বেসিক ইত্যাদি।	**উদাহরণ ঃ** সি++, জাভা ইত্যাদি।

৩.২ ট্র্যাডিশনাল প্রোগ্রামিং এর অসুবিধা (Drawbacks of traditional programming) ঃ

নিম্নে ট্র্যাডিশনাল প্রোগ্রামিং-এর অসুবিধাসমূহ আলোচনা করা হল। যথা ঃ

- ট্র্যাডিশনাল প্রোগ্রামিং-এ হাজার হাজার লাইনের বড় প্রোগ্রাম লিখা সহজ কিন্তু পরবর্তীতে এর কোডিং করা বেশ কষ্টকর।
- বড় বড় প্রোগ্রামের ক্ষেত্রে কোন ফাংশনের জন্য কোন ডাটা ব্যবহৃত হচ্ছে তা identify করা জটিল ও সময় সাপেক্ষ ব্যাপার।
- এতে বিভিন্ন ফাংশনের জন্য গ্লোবাল ডাটা ব্যবহার করা হয় যাতে একই ডাটা বিভিন্ন ফাংশন এর জন্য ব্যবহৃত হয়। ফলে সিস্টেমের security হ্রাস পায়।
- প্রয়োজনে যে কোন সময় প্রোগ্রামে অতিরিক্ত ডাটা বা ফাংশন যোগ করা যায় না। এ জন্য পুরো প্রোগ্রাম টিকে change করতে হতে পারে।
- এ ধরনের প্রোগ্রামিং-এ ভুল-ত্রুটি নির্ণয় ও এদের সমাধান করা অনেক কঠিন।
- ট্র্যাডিশনাল প্রোগ্রামিং-এ একটি বড় সমস্যাকে কতগুলো ক্ষুদ্র ক্ষুদ্র অংশে ভাগ করা হয় এবং প্রতিটি ভাগের জন্য এক বা একাধিক ফাংশন বর্ণনা করা হয় যাদেরকে একত্রিত করে মূল সমস্যার সমাধান করা হয়। তাই এই প্রক্রিয়া বেশ জটিল।
- এই পদ্ধতিতে বড় বড় প্রোগ্রাম লিখতে গিয়ে data abstraction, code reusablity ইত্যাদি বিষয়ে সমস্যায় পড়তে হয়।
- এই পদ্ধতিতে প্রোগ্রামের মেইনটেন্যান্স করা অনেক কঠিন কাজ।
- সর্বোপরি এতে ব্যবহৃত ফাংশনসমূহ action oriented যা বাস্তব জীবনের সাথে সম্পর্কিত নয়।

৩.৩ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর বিভিন্ন টার্মসমূহ (Terms used in OOP-Object, Class, Data abstraction, Encapsulation, Inheritance, Polymorphism, Message Passing and Dynamic Binding) ঃ

অবজেক্ট (Object) ঃ অবজেক্ট বলতে সাধারণত কোন বস্তুকে বুঝায়। যেমন– মানুষ, বই, খাতা, পেন্সিল, কলম, পৃথিবী ইত্যাদি। প্রতিটি অবজেক্টের কিছু সাধারণ (Common) বৈশিষ্ট্য থাকে আবার কিছু নিজস্ব বৈশিষ্ট্যও থাকে যাদের মাধ্যমে একটি অবজেক্টকে অন্য একটি অবজেক্ট থেকে আলাদা করা যায়।

অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং (OOP) এর ক্ষেত্রে ক্লাস (Class) এ ব্যবহৃত ভেরিয়েবল (Variable)সমূহের প্রত্যেকটিকে এক একটি অবজেক্ট (Object) বলা হয়।

উদাহরণস্বরূপ বলা যায়, যদি প্রোগ্রামে Student নামে কোন Class থাকে তবে ঐ Class এ ব্যবহৃত ভেরিয়েবলসমূহ (Name, roll, session, GPA ইত্যাদি) এক একটি অবজেক্ট।

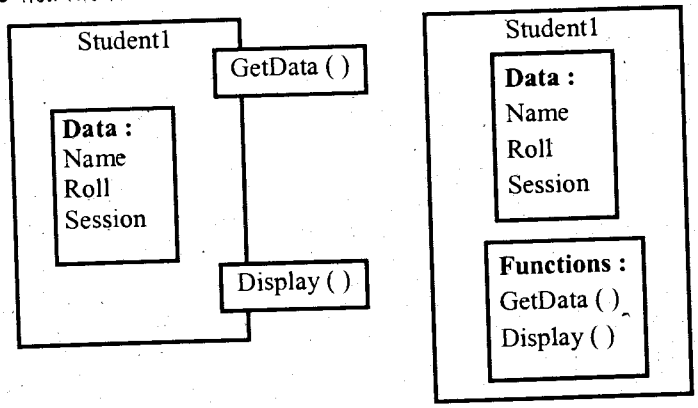
মূলত একটি ক্লাস তৈরির পর প্রোগ্রামে সেই ক্লাস এর কোন ভেরিয়েবল ডিক্লেয়ার করে প্রোগ্রামে মূল অবজেক্টকে উপস্থাপন করা হয়। তাই বলা যায় প্রোগ্রামে ক্লাস (Class) হল কোন অবজেক্টের লজিক্যাল রিপ্রেজেন্টেশন (Logical reprentation) আর সেই ক্লাস এর ভেরিয়েবল হল সেই অবজেক্টের ফিজিক্যাল রিপ্রেজেন্টেশন (Physical representation)।

প্রতিটি অবজেক্টে কিছু ডাটা এবং ঐ ডাটা এক্সেস করার জন্য কিছু ফাংশন থাকে। সাধারণত বাহিরের অন্য কোন ফাংশন দ্বারা এই ডাটা এক্সেস করা যায় না। ফাংশন গুলোর মাধ্যমে একটি অবজেক্ট অন্য এক বা একাধিক অবজেক্টের সাথে যোগাযোগ রক্ষা করে। অবজেক্ট মেমোরিতে স্থান দখল করে এবং প্রতিটি অবজেক্টের একটি মেমোরি অ্যাড্রেস থাকে। প্রতিটি অবজেক্ট মেমোরিতে আলাদা স্থানে থাকে। যেহেতু প্রতিটি অবজেক্টের মেমোরি অ্যাড্রেস আলাদা, সেহেতু একই প্রোগ্রামে একাধিক অবজেক্ট কিংবা একই অবজেক্ট একাধিক প্রোগ্রামে ব্যবহার করলেও একটি অবজেক্ট অন্য একটির সাথে মিশে যায় না। প্রোগ্রাম নির্বাহের সময় অবজেক্টসমূহ ম্যাসেজ আদান-প্রদানের মাধ্যমে একে অপরের সাথে যোগাযোগ করে।

প্রোগ্রামে প্রথমে কোন ক্লাস (Class) ডিক্লেয়ার করতে হয়। অতঃপর ঐ ক্লাসের অধীনে এক বা একাধিক অবজেক্ট ঘোষণা করা যায়। যেমন- Student নামক ক্লাসের Student1, Student2, Student3, ---, StudentN নামের অবজেক্টগুলোকে নিম্নলিখিতরূপে ঘোষণা করা যায় ঃ

```
Student, Student1, Student2, Student3, --- , StudentN;
```

একই অবজেক্টের ডাটা এবং ফাংশনসমূহের ক্রিয়াকলাপ বিভিন্নজন বিভিন্নভাবে বর্ণনা করেন। নিম্নে Student1 নামের অবজেক্টটিকে দু'টি উপায়ে রিপ্রেজেন্ট করে দেখানো হল।



চিত্র ঃ অবজেক্ট রিপ্রেজেন্টেশন

ক্লাস (Class) ঃ সাধারণভাবে ক্লাস (Class) হল একই ধরনের কতগুলো অবজেক্টের সমন্বিত নাম। যেমন– 'পাখিকে' একটি ক্লাস বিবেচনা করলে 'কাক, কোকিল, টিয়া, ময়না, বুলবুলি' ইত্যাদি ঐ ক্লাসের এক একটি 'অবজেক্ট (Object)। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর ক্ষেত্রেও ক্লাস একই অর্থ বহন করে। কাজেই ক্লাস (Class) হচ্ছে এক ধরনের ইউজার ডিফাইন্ড ডাটা টাইপ যাতে একই শ্রেণির বা গোত্রের এক বা একাধিক অবজেক্ট থাকে। এটি প্রোগ্রামে বিল্ট-ইন ডাটা টাইপের ন্যায় কাজ করে। একটি ক্লাসে বিল্ট-ইন ডাটা টাইপের কয়েকটি ভেরিয়েবল এবং ঐ ভেরিয়েবলগুলোকে এক্সেস (Access) করার জন্য কতগুলো ফাংশন থাকে। প্রয়োজন অনুযায়ী প্রোগ্রামে কোন ক্লাস ঘোষণা করা হয়। প্রোগ্রামে এক বা একাধিক ক্লাস থাকতে পারে। পূর্বে তৈরি করা কোন ক্লাসকে কোনরূপ পরিবর্তন ছাড়া কিংবা নতুন কোন বৈশিষ্ট্য যোগ করে অন্য কোন প্রোগ্রামে ব্যবহার করা যায়।

ক্লাস ঘোষণার জন্য class কীওয়ার্ড-এর পর ঐ ক্লাসের নাম দিতে হয়। ক্লাস বডি দ্বিতীয় বন্ধনী '{ }' দ্বারা সীমাবদ্ধ থাকে এবং সেমিকোলন (;) দ্বারা শেষ হয়। ক্লাস বডি'তে ক্লাসের মেম্বার ভেরিয়েবল এবং মেম্বার ফাংশন থাকে। ক্লাসের ভৈরিয়েবলগুলোকে মেম্বর ভেরিয়েবল এবং ফাংশনগুলোকে মেম্বার ফাংশন বা সংক্ষেপে মেথড বলা হয়। নিম্নে Student নামের একটি নমুনা ক্লাসের ফরম্যাট দেখান হল ঃ

```
class Student
{
private :   / / private accessibility mode
    long Roll ;
    char Name [20] ;
    float Marks ;
public :            / / public accessibility mode
    void GetData ( )
    {
    / / Function Body
    }
    void Display ( )
            {
             / / Function Body
            }
} ;         / / End of class Decleration
```

ইনহেরিটেন্স (Inheritance) ঃ ইংরেজি Inheritance শব্দের অর্থ "উত্তরাধিকার সূত্রে পাওয়া।" কোন বড় প্রোগ্রামে অনেক সময় প্রায় একই কার্য সম্পাদন করার জন্য কিছু প্রোগ্রামাংশ বার বার লেখার প্রয়োজন হয়। তবে একই প্রোগ্রামে নির্ভুল, পরীক্ষিত এবং এক বা একাধিকবার ব্যবহৃত কোন প্রোগ্রামাংশ বা class নতুন করে তৈরি না করে পুনঃ পুনঃ ব্যবহার করা অনেক সুবিধাজনক। এতে সময় ও খরচ কিংবা প্রোগ্রামের ঝামেলা কমানোর সাথে সাথে প্রোগ্রামের গ্রহণ যোগ্যতা ও বাড়ে। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং পদ্ধতিতে একটা ক্লাস প্রয়োজনে আরেকটা ক্লাস কে inherit করার এই প্রক্রিয়াকে ইনহেরিটেন্স (inheritance) বলে।

অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর ক্ষেত্রে যখন কোন ক্লাস অন্য একটি ক্লাস এর এক বা একাধিক বৈশিষ্ট্য (attribute, behaviours) ইনহেরিট করে তখন প্রথমোক্ত ক্লাসটিকে ডিরাইভড ক্লাস (Derived class) আর যেই ক্লাস থেকে বৈশিষ্ট্য ইনহেরিট করা হবে তাকে বেস ক্লাস (Base class) বলে।

নিম্নে Student ক্লাসকে বেস ক্লাস ধরে ইনহেরিটেন্স প্রক্রিয়ায় Result নামের ডিরাইভড ক্লাস গঠনের ফরম্যাট দেখানো হল–

```
class Student
  {
private :
        long Roll ;
protected :
        char Name [20] ;
public :
        void GetData () ;
        void Display () ;
        } ;
class Result : public Student
private :
        float Marks ;
public :
        void Getmarks () ;
        void Showmarks () ;
    } ;
```

এক্ষেত্রে Result ক্লাসের নতুন অবস্থা হবে ঃ

```
class Result
        {
        private :
             float Marks ;
             protected :
             char Name [20] ;   / / Inherited from Student Class
        public :
             void GetData ();    / / Inherited from Student Class
             void Display () ;   / / Inherited from Student Class
             void Getmarks () ;
             void showMarks () ;
        } ;
```

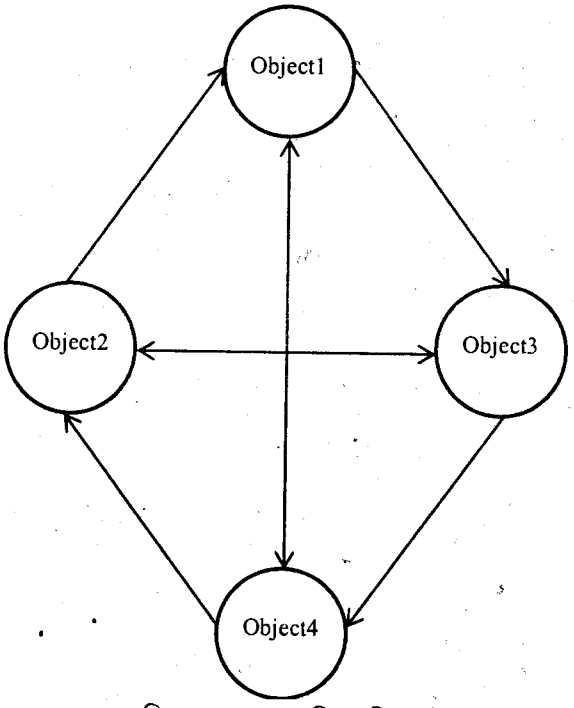
ডাটা অ্যাবস্ট্রাকশন (Data Abstraction) ঃ ডাটা অ্যাবস্ট্রাকশন অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর একটি অত্যাবশ্যকীয় বৈশিষ্ট্য যা ক্লাসে বিদ্যমান থাকে। ডাটা অ্যাবস্ট্রাকশন হচ্ছে এমন এক ধরনের ডাটা টাইপ বর্ণনা কৌশল যেখানে কিছু ডাটা থাকে এবং ঐ ডাটা এক্সেস করার জন্য কতগুলো ফাংশন থাকে। তবে ঐ টাইপের কোন ভেরিয়েবলের ডাটা তার নিজস্ব ফাংশনের সাহায্য ছাড়া অন্য কোন ভাবে access করা যায় না। যেহেতু ক্লাস ও ডাটা অ্যাবস্ট্রাকশন-এর ধারণা একই রকম তাই তারা Abstract Data Type (ADT) নামে ও পরিচিত। হাই লেভেল সিস্টেম ডিজাইনের ভিত্তি হিসাবে Data Abstraction ব্যবহৃত হয়। Multi Module প্রসেসিং সিস্টেমে Abstract ডাটা টাইপ এর অবজেক্ট ব্যবহার সুবিধাজনক।

এনক্যাপসুলেশন (Encapsulation) ঃ এনক্যাপসুলেশন (Encapsulation) হচ্ছে এমন একটি প্রক্রিয়া যার মাধ্যমে কোন অবজেক্টের ডাটা ও ফাংশনসমূহকে একত্রিত করে ক্লাস তৈরি করা হয়। এর মাধ্যমে ডাটা ও ফাংশনসমূহ বাইরের অবাঞ্ছিত এক্সেস হতে রক্ষা পায়। এনক্যাপসুলেশনের কারণে অবজেক্টের ডাটা ও ফাংশনসমূহ একটি একক ইউনিটের মত ক্রিয়া করে। ফলে ডাটাকে আলাদা করা যায় না কিংবা ডাটাকে তার ফাংশন থেকে আলাদা করা যায় না এবং ক্লাসে কোন অবজেক্ট ঘোষণা করলে তাতে ঐ ক্লাসের সকল বৈশিষ্ট্য অক্ষুণ্ন থাকে। তাই এনক্যাপসুলেশন অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর একটি অত্যাবশ্যকীয় বৈশিষ্ট্য হিসেবে বিবেচিত।

পলিমরফিজম (Polymorphism) ঃ পলিমরফিজম অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং-এর অপর একটি গুরুত্বপূর্ণ বৈশিষ্ট্য। পলিমরফিজম দুটো গ্রিক শব্দ পলি এবং মরফিজম-এর সমন্বয়ে গঠিত, যাদের অর্থ যথাক্রমে বহু এবং রূপ। সুতরাং পলিমরফিজম বলতে বুঝায় বহুরূপ। অর্থাৎ ভিন্ন ভিন্ন অবস্থায় একই অবজেক্টের ভিন্ন ভিন্ন ফল প্রদান করা। যেমন– প্লাস অপারেটর (+), যা সাধারণত দুইটি নিউমেরিক সংখ্যার যোগফল হিসেবে তৃতীয় একটি নিউমেরিক সংখ্যা প্রদান করে। তবে এই অপারেটর বিভিন্ন অপার‍্যান্ডের জন্য বিভিন্ন ফলাফল প্রদান করতে পারে। যেমন– দুইটি স্ট্রিং-এর ফলাফল হিসেবে তৃতীয় একটি স্ট্রিং প্রদান করতে পারে। মূলত অপারেটর ওভারলোডিং-এর মাধ্যমে এই কাজ করা হয়। অপারেটর ওভারলোডিং ব্যবস্থায় একই অপারেটর ভিন্ন ভিন্ন ডাটা টাইপ অপারেন্ডের জন্য ভিন্ন ভিন্ন ফলাফল প্রদান করতে পারে। অপারেটর ওভারলোডিং একটি কম্পাইল টাইম পলিমরফিজম। ফাংশন ওভারলোডিং নামে অপর একটি কম্পাইল-টাইম পলিমরফিজম আছে। ফাংশন ওভারলোডিং-এর মাধ্যমে একই সাথে একই নামে বিভিন্ন আরগুমেন্ট টাইপ এবং আরগুমেন্ট সংখ্যাবিশিষ্ট দুই বা ততোধিক ফাংশন ব্যবহার করা যায়। ওভারলোডেড ফাংশনগুলো মেমোরির কিছু অংশ শেয়ার করে ফলে প্রোগ্রাম নির্বাহন দ্রুত হয়। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং-এ রান-টাইম পলিমরফিজম নামে অপর এক প্রকার পলিমরফিজম বিদ্যমান এবং পলিমরফিজম বলতে সাধারণত রান-টাইম পলিমরফিজমকেই বুঝায়। রান-টাইম পলিমরফিজম-এর মাধ্যমে কম্পাইলার প্রোগ্রাম চলাকালীন সময় বেস ক্লাস এবং ডিরাইভড ক্লাসে একই প্রোটোটাইপ বিশিষ্ট দুই বা ততোধিক মেম্বার ফাংশনের মধ্য থেকে উপযুক্ত ফাংশন নির্বাচন করতে পারে। এ প্রক্রিয়ায় ভিন্ন ভিন্ন ক্লাসের কতকগুলো অবজেক্ট একই গ্রুপে বর্ণনা করা হয় এবং কম্পাইলার অবজেক্টগুলোর রেফারেন্স টাইপের ভিত্তিতে উপযুক্ত ফাংশন কল করে স্ব স্ব কার্য নির্বাহ করে।

ম্যাসেজ পাসিং (Message passing) ঃ অবজেক্ট অরিয়েন্টেড সিস্টেমে অবজেক্টগুলো প্রয়োজনে পরস্পরের মধ্যে সংযোগ রক্ষা করে। সাধারণভাবে বলা চলে, মানুষ যেভাবে পরস্পরের মধ্যে যোগাযোগ করে অবজেক্টগুলো অনেকটা সেভাবেই পরস্পরের মধ্যে সংযোগ রক্ষা করে। মূলত অবজেক্টগুলো তাদের মেম্বার ফাংশনের মাধ্যমে এক অপরের সাথে সংযোগ রক্ষা করে। কোন অবজেক্টের ফাংশন কেবল ঐ অবজেক্টের নিজস্ব ডাটা ছাড়া সরাসরি অন্য অবজেক্টের ডাটা এক্সেস করতে পারে না। যদি কখনও কোন অবজেক্টকে অপর কোন অবজেক্টের সাথে সংযোগ করতে হয় তবে প্রথম অবজেক্ট তার নিজস্ব মেম্বার ফাংশনের মাধ্যমে দ্বিতীয় অবজেক্টকে অনুরোধ পাঠায়। তারপর দ্বিতীয় অবজেক্ট তার ক্ষমতানুযায়ী নিজস্ব মেম্বার ফাংশনের মাধ্যমে অনুরোধের ফলাফল পাঠায়। অবজেক্টগুলোর মধ্যে এরূপ সংযোগ রক্ষার প্রক্রিয়াকে ম্যাসেজ পাসিং বলা হয়। এক্ষেত্রে অনুরোধকারী অবজেক্টকে ক্লায়েন্ট অবজেক্ট এবং ফলাফল প্রদানকারী অবজেক্টকে সার্ভার অবজেক্ট বলা হয়। আবার একই সাথে সার্ভার অবজেক্টও ক্লায়েন্ট অবজেক্টকে অনুরোধ পাঠাতে পারে। তবে একটি অবজেক্ট যতক্ষণ সক্রিয় থাকে ঠিক ততক্ষণই ম্যাসেজ পাসিং-এ সাড়া দিতে পারে।

নিম্নের চিত্রের মাধ্যমে একাধিক অবজেক্টের মধ্যে ম্যাসেজ পাসিং প্রক্রিয়া দেখানো হল ঃ



চিত্র ঃ ম্যাসেজ পাসিং প্রক্রিয়া

ম্যাসেজ পাসিং এর সময় মূলত তিনটি শর্ত পূরণ করতে হয়। শর্ত তিনটি হচ্ছে-

- অবজেক্টের নাম
- মেথডের নাম ও
- ইনফরমেশন

উদাহরণ ঃ employee. salary (name);

এখানে, 'employee' হচ্ছে অবজেক্টের নাম, 'salary' মেথডের বা ফাংশনের নাম আর 'name' হচ্ছে ইনফরমেশন।

ডাইনামিক বাইন্ডিং (Dynamic binding) ঃ ডাইনামিক অর্থ পরিবর্তনশীল। স্ট্যাটিক বাইন্ডিং প্রক্রিয়ায় ফাংশন ওভারলোডিং-এর ক্ষেত্রে ফাংশনের আরগুমেন্টগুলোর ডাটা টাইপ ও সংখ্যা এবং অপারেটর ওভারলোডিং-এর ক্ষেত্রে অপারেশনে ব্যবহৃত অপার্যান্ডগুলোর ডাটা টাইপের মাধ্যমে অবজেক্ট (ফাংশন, অপার্যান্ড, অপারেটর ইত্যাদি) গুলোর মধ্যে স্থির পার্থক্য বিদ্যমান থাকে। এসব পার্থক্য কম্পাইলার প্রোগ্রাম কম্পাইল করার সময় ধরতে পারে। কিন্তু যদি দুই বা ততোধিক ক্লাসে (যেমন– বেস ক্লাস এবং ডিরাইভড ক্লাসে) একই নাম, একই প্রটোটাইপ এবং আরগুমেন্ট টাইপ ও সংখ্যাবিশিষ্ট দুই বা ততোধিক আলাদা ফাংশন থাকে এবং প্রোগ্রাম পয়েন্টার বার বার এক ফাংশন থেকে অন্য ফাংশনে পরিবর্তন করতে হয় তবে কম্পাইল করার সময় কম্পাইলার কর্তৃক সঠিক ফাংশন নির্বাচন করা সম্ভব নাও হতে পারে। এক্ষেত্রে প্রোগ্রাম নির্বাহের সময় সঠিক ফাংশন নির্বাচনের কোন বিকল্প থাকে না। মূলত ফাংশনগুলোর ডাইনামিক রেফারেন্সের ভিত্তিতে প্রোগ্রাম নির্বাহের সময় উপযুক্ত ক্লাসের অবজেক্ট নির্বাচনের মাধ্যমে সঠিক ফাংশন নির্বাচন করা হয়, কিংবা পয়েন্টার এক ফাংশন থেকে অপর ফাংশনে স্থানান্তর করা হয়। এ প্রক্রিয়াকে ডাইনামিক বাইন্ডিং বলা হয়। ডাইনামিক বাইন্ডিং প্রক্রিয়ায় একাধিক রেফারেন্স বিশিষ্ট দুই বা ততোধিক ফাংশন থেকে কোন ফাংশন নির্বাচন প্রক্রিয়া ফাংশনগুলোর রেফারেন্স অ্যাড্রেসের উপর নির্ভর করে।

৩.৪ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজের তালিকা (Name of some OOP languages) ঃ

অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ বলতে নির্দিষ্ট কোন প্রোগ্রামিং ল্যাংগুয়েজকে বুঝায় না। মূলত অবজেক্ট অরিয়েন্টেড টেকনোলজির সুবিধা প্রদানকারী যেকোন ভাষাকে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ বলা হয়। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজের প্রধান বৈশিষ্ট্যগুলো হচ্ছে- ক্লাস (Class), অবজেক্ট (Object), ইনহেরিটেন্স (Inheritance) ও পলিমরফিজম ইত্যাদি। উল্লিখিত বৈশিষ্ট্যসম্পন্ন অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজের তালিকা নিম্নরূপ ঃ

- স্মলটক (Small Talk)
- সি++ (C++)
- টার্বো প্যাসকাল (Turbo Pascal)
- হাইপার টক (Hyper Talk)
- জাভা (Java)
- ভিজুয়াল বেসিক (Visual Basic)
- ভিজুয়াল সি++ (Visual C++)
- ভিজুয়াল জে++ (Visual J++) ইত্যাদি।

স্মলটক (Small talk) ঃ Small Talk হচ্ছে সর্ব প্রথম OOP language। ১৯৭০ সালে ক্যালিফোর্নিয়ার "Xerox palo Alto Research Center" এর কম্পিউটার বিজ্ঞানী "Alan Kay" Small Talk আবিষ্কার করেন। এতে কেবল Text লিখার জন্য Keyboard এবং বাকি সব কাজের জন্য mouse ব্যবহার করা হত।

সি++ (C++) ঃ সি++ হচ্ছে বাস্তবসম্মত সমস্যা সমাধানের জন্য বহুল ব্যবহৃত একটি অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ। ১৯৮০ সালে যুক্তরাষ্ট্রের এটিএন্ডটি বেল ল্যাবরেটরি ও জর্ন স্ট্রাউস্ট্রাপ এর নেতৃত্বে সিমুলা 67 এবং C ভাষা দিয়ে শুরু হওয়া গবেষণার ফলে ১৯৮৩ সালে নতুন এক ভাষার জন্ম হয় যার নাম সি++ (C++)। এতে C ভাষার প্রধান প্রধান বৈশিষ্ট্যসমূহের সাথে বাড়তি (++, Plusplus) কিছু বৈশিষ্ট্য যেমন- সিমুলা 67 ভাষার ক্লাসের ধারণা যোগ করা হয়েছে বলেই এর নামকরণ সি++। সি++, সি ভাষার মত প্রসিডিউর প্রোগ্রামিং এর সুবিধা প্রদান করে এবং একই সাথে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং পদ্ধতির ব্যবহার ও অনুমোদন করে। তাই সি++ কে হাইব্রিড প্রোগ্রামিং ল্যাংগুয়েজও বলা চলে। টেক্সট এডিটর তৈরি, কম্পাইলার ডিজাইন, ডাটাবেজ হ্যান্ডলিং, কমিউনিকেশন সিস্টেম ডিজাইন, ডিস্ট্রিবিউটেড সিস্টেম ডিজাইন, রিয়েল টাইম সিস্টেম ডিজাইনসহ যেকোন এপ্লিকেশন সফটওয়্যার তৈরিতে সি++ ভাষার জুড়ি নেই। তাছাড়া সি++ প্রোগ্রামসমূহ সহজেই পরিচালনা করা যায় এবং প্রয়োজনে পরিবর্তন ও পরিবর্ধন করা যায়।

টার্বো প্যাসকাল (Turbo Pascal) ঃ একে object oriented pascal ও বলা হয়। এটি ১৯৮৪ সালে 'Borland International' এর "Philippe Kahn" ডিজাইন করেন। ইহা প্রোগ্রামের বিভিন্ন মডিউল কিংবা অবজেক্টকে পরিবর্তন করে মডিফাই করার সুযোগ প্রদান করে।

হাইপার টক (Hyper Talk) ঃ Hyper Talk, Hyper card এর জন্য ব্যবহৃত ল্যাংগুয়েজ। Hyper card হচ্ছে এমন এক ধরনের electronic card বা card সমষ্টি যাতে ডাটা বা ইনফরমেশন স্টোর করা যায়। Hyper Talk এমন এক ধরনের language যা OOP principles মেনে চলে এবং Hyper Card stack সমূহকে manipulate করার জন্য Hyper card program ব্যবহৃত হয়।

জাভা (Java) ঃ বর্তমান সময়ে সর্বাধুনিক ও সর্বশ্রেষ্ঠ পূর্ণাঙ্গ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ হিসেবে স্বীকৃতি পাওয়া ল্যাংগুয়েজের নাম হচ্ছে জাভা (Java)। যুক্তরাষ্ট্রের Sun Microsystem কোম্পানির গবেষক জেমস গসলিং (James gosling) এর নেতৃত্বে একদল বিশেষজ্ঞের দীর্ঘ ১৮ মাসের সম্মিলিত গবেষণার ফলে ১৯৯১ সালে জাভা ভাষা আত্মপ্রকাশ করে। তবে জাভার প্রথম ভার্সন প্রকাশিত হয় ১৯৯৬ সালে। মূলত সি ও সি++ ল্যাংগুয়েজের সমস্যা সৃষ্টিকারী বৈশিষ্ট্যগুলো বাদ দিয়ে, অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর নতুন অনেক বৈশিষ্ট্য যোগ করে এ ভাষা ডেভেলপ করা হয়েছে। তাই জাভা একটি সহজ, সাবলীল, বিশ্বস্ত, গ্রহণযোগ্য ও ইউজার ফ্রেন্ডলি ল্যাংগুয়েজ। তাছাড়া জাভাই প্রথম কোন অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ যা নির্দিষ্ট কিছু হার্ডওয়্যার ও সফটওয়্যারের গণ্ডির মধ্যে সীমাবদ্ধ নয়। ফলে এক কম্পিউটারে কিংবা অপারেটিং সিস্টেমে লেখা জাভা প্রোগ্রাম অন্য কম্পিউটার কিংবা অপারেটিং সিস্টেমে রান করা সম্ভব। জাভা পুরোপুরিভাবে ইন্টারনেট ও ওয়ার্ল্ড ওয়াইড ওয়েব (WWW) প্রযুক্তির সাথে সম্পর্কিত একটি ভাষা। ফলে ইন্টারনেট ও নেটওয়ার্ক প্রোগ্রামিং এর জনপ্রিয়তা বৃদ্ধির সাথে সাথে জাভার জনপ্রিয়তাও উত্তরোত্তর বৃদ্ধি পাচ্ছে।

ভিজুয়াল বেসিক, ভিজুয়াল সি++, ভিজুয়াল জে++ (Visual basic, Visual C++, Visual J++) ঃ ভিজুয়াল বেসিক, ভিজুয়াল সি++, ভিজুয়াল জে++ যথাক্রমে বেসিক, সি++ এবং জাভা প্রোগ্রামিং ল্যাংগুয়েজের ভিজুয়াল ভার্সন। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজের এসব ভার্সনগুলোতে মাউস ক্লিক করে অথবা মাউস ড্রাগ করে কোন অবজেক্ট বা ক্লাস অতি সহজেই অপর কোন প্রোগ্রামে ব্যবহার করা যায়। ফলে প্রোগ্রামের আকার ছোট হয় এবং গ্রহণযোগ্যতা বৃদ্ধি পায়।

৩.৫ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর সুবিধা-অসুবিধা (Benifits and drawbacks of OOP) ঃ

অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর সুবিধা (Benifits of OOP) ঃ প্রোগ্রাম ডিজাইনার এবং সিস্টেম ব্যবহারকারী উভয়ের জন্য অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং কতকগুলো সুবিধা প্রদান করে। এ অবস্থায় কম রক্ষণাবেক্ষণ খরচে, অল্প সময়ে, অল্প পরিশ্রমে উন্নত মানের সফটওয়্যার নির্মাণ করা সম্ভব। বর্তমানে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এত জনপ্রিয় যে প্রোগ্রামিং বলতে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিংকেই বুঝায় এবং অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ছাড়া সিস্টেম ডিজাইন চিন্তাই করা যায় না। এমন কোন সমস্যা নেই যা অবজেক্ট অরিয়েন্টেড পদ্ধতিতে সমাধান করা যায় না। প্রোগ্রামিং-এর গতানুগতিক ধারা পাল্টে দিয়েছে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং। মোট কথা অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং-এর সুবিধা বলে শেষ করা যাবে না। নিম্নে এরূপ কয়েকটি সুবিধা উল্লেখ করা হল ঃ

- বাস্তব ভিত্তিক সিস্টেম ডিজাইনে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজের কোন বিকল্প নেই। কারণ এতে বাস্তবতার আলোকে খুব সহজে যে কোন সমস্যার সমাধান করা সম্ভব।
- এতে বড় কোন সমস্যাকে ছোট ছোট অংশে ভাগ করে ভাগগুলির সমাধান একত্রিত করে সমস্যার সমাধান করা হয়। ফলে ভুল-ত্রুটি নির্ণয় সুবিধাজনক।
- ডাটা হাইডিং এর মাধ্যমে প্রোগ্রামের ডাটাকে কিংবা প্রোগ্রামের security রক্ষা করা যায়।
- Inheritance system প্রয়োগ করে পুরাতন ক্লাসের পরিধি বাড়িয়ে নতুন ক্লাস তৈরি করা যায়।
- প্রয়োজনে যে কোন সময় প্রোগ্রামে অতিরিক্ত ডাটা বা ফাংশন যোগ করা যায়। এজন্য পুরো প্রোগ্রাম পরিবর্তন করতে হয় না।

- সমস্যার ধরন অনুযায়ী এ প্রোগ্রামিং নিজস্ব ডাটা টাইপ তৈরি করে এবং এতে যে কোন সংখ্যক object ব্যবহার করা যায়।
- প্রয়োজনে একই প্রোগ্রামে একাধিক ডাটা টাইপের object ব্যবহার করা যায় এবং একই object কে ভিন্ন ভিন্ন রূপে ব্যবহার করা যায়।
- এই পদ্ধতিতে Multiple প্রোগ্রামার সিস্টেমে প্রোগ্রাম ডিজাইন করা যায়।
- ইহা Networking program এর সুবিধা প্রদান করে।
- প্রোগ্রাম টেস্টিং ও ডিবাগিং করা সহজ তাই প্রোগ্রামের জটিলতা হ্রাস পায় এবং
- প্রোগ্রামের আকার বেড়ে গেলেও এই পদ্ধতিতে প্রোগ্রামের রক্ষণাবেক্ষণ, পরিবর্তন, পরিবর্ধন, পরিমার্জন করা সহজ।

অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর অসুবিধা (Drawbacks of OOP) ঃ বাস্তব ভিত্তিক সিস্টেম ডিজাইনে অবজেক্ট-অরিয়েন্টেড প্রোগ্রামিং-এর কোন বিকল্প নেই। তবে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ব্যবহারে কিছু সমস্যা আছে। যেমন–

- প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং-এর মত অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং পদ্ধতি অনেকটা অপরিচিত।
- অবজেক্ট অরিয়েন্টেড টেস্টিং, ডিবাগিং, মেইনটেনন্স পদ্ধতি কিছুটা জটিল।
- ইনহেরিটেড অবজেক্টের ক্ষেত্রে মাল্টিপল ক্লাস হায়ারারকির জন্য বিভিন্ন ক্লাসের মেম্বার ভেরিয়েবল এবং মেম্বার ফাংশনকে প্রতিনিয়ত মেমোরির বিভিন্ন অবস্থান এবং সোর্স ফাইল থেকে ডাটা আদান-প্রদান করতে হয়। এজন্য সিস্টেমে ডাটার নির্দিষ্ট কোন গতিপথ থাকে না।
- ম্যাসেজ পাসিং-এর ক্ষেত্রে এক অবজেক্টের ফাংশন থেকে অপর কোন অবজেক্টের ফাংশনে যোগাযোগের সময় অতিরিক্ত সময় ব্যয় হয়।
- পলিমরফিজম অর্থাৎ ডাইনামিক বাইন্ডিং এর ক্ষেত্রে প্রোগ্রাম কাউন্টার বারবার এক ফাংশন থেকে অন্য ফাংশনে পরিবর্তন করতে হয়, ফলে প্রসেসিং টাইম বেশি।

৩.৬ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর ব্যবহার (Application of OOP) ঃ

অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং আজ সর্বজন স্বীকৃত। বাস্তব ভিত্তিক যেকোন সমস্যা সমাধানে এর কোন জুড়ি নেই। কার্যকর ও নতুন নতুন বৈশিষ্ট্যের কারণে এই প্রোগ্রামিং এর চাহিদা ব্যাপক এবং আজকে এটি সবচেয়ে জনপ্রিয় একটি প্রোগ্রামিং পদ্ধতি। কম্পিউটার সিস্টেমে এমন কোন ক্ষেত্র নেই যেখানে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর অবদান নেই। নিম্নে OOP এর ব্যবহার সমূহ আলোচনা করা হল। যথা ঃ

- রিয়েল টাইম সিস্টেম ডিজাইনের ক্ষেত্রে
- সিমুলেশন ও মডেলিং সিস্টেমে
- অবজেক্ট অরিয়েন্টেড ডাটাবেজ সিস্টেমে
- হাইপার টেক্সট, হাইপার মিডিয়া ও এক্সপার্ট টেক্সট সিস্টেমে
- আর্টিফিসিয়াল ইন্টেলিজেন্স ও এক্সপার্ট সিস্টেমে
- নিউরাল নেটওয়ার্কস ও প্যারালাল প্রোগ্রামিং এ
- কম্পাইলার ডিজাইন
- কমিউনিকেশন সিস্টেম ডিজাইন
- ডিস্ট্রিবিউটেড সিস্টেম ডিজাইন
- ডিসিশন সাপোর্ট ও অফিস অটোমেশন সিস্টেম
- সিম, ক্যাম, ক্যাড (CIM, CAM, CAD) ইত্যাদি সিস্টেম এবং
- সফটওয়্যার ইন্ডাস্ট্রিতে সফটওয়্যার ডিজাইন ও এনালাইসিস প্রভৃতি ক্ষেত্রে।

অনুশীলনী-৩

অতি সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। মেশিন ল্যাংগুয়েজ কাকে বলে?

উত্তরঃ যে ভাষায় সমস্যা সমাধানের জন্য বাইনারি সংখ্যা (0,1) ব্যবহার করে নির্দেশ (Instruction) সাজিয়ে প্রোগ্রাম রচনা করা হয়, তাই মেশিন ল্যাংগুয়েজ (Machine language)।

২। এসেম্বলি ল্যাংগুয়েজ কাকে বলে?

উত্তরঃ বিশেষ সংক্ষিপ্ত সংকেত বা নেমোনিক (ADD, SUB, MUL, DIV, LOAD, MOV, MVI, JMP, JNZ, INC ইত্যাদি) ব্যবহার করে যে ভাষায় প্রোগ্রাম রচনা করা হয় তাকে এসেম্বলি ল্যাংগুয়েজ (Assembly language) বলে। যেমন- MASM, TASM ইত্যাদি।

৩। প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং বলতে কী বোঝায়?

উত্তরঃ হাই লেভেল ল্যাংগুয়েজ যেমন ঃ COBOL, ForTran, BASIC, C ইত্যাদি ব্যবহার করে যে প্রোগ্রাম তৈরি করা হয় তাকে স্ট্রাকচার্ড বা প্রসিডিউর অরিয়েন্টেড প্রোগ্রাম বলে।

৪। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং বলতে কী বোঝায়?

উত্তরঃ প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজে ডাটার চেয়ে ফাংশনের গুরুত্ব বেশি এবং যখনই এ ধরনের প্রোগ্রামের আকার বেড়ে যায় তখন এর ভুল-সংশোধন, পরিবর্তন, পরিবর্ধন, পরিমার্জন ও রক্ষণাবেক্ষণ ইত্যাদি কঠিন হয়ে পড়ে। এ সকল সমস্যা সমাধানের নিমিত্তে যে ধরনের প্রোগ্রামিং পদ্ধতির আবির্ভাব ঘটে তাকে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং বলে।

৫। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ বলতে কী বোঝায়?

উত্তরঃ আবজেক্ট অরিয়েন্টেড টেকনোলজির সুবিধা সম্পন্ন (যেমন– ইনহেরিটেন্স, পলিমরফিজম, ডাটা অ্যাবস্ট্রাকশন, এনক্যাপসুলেশন, ম্যাসেজ পাসিং) যে কোন ভাষাকে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ বলে।

৬। কয়েকটি অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজের নাম লেখ।

উত্তরঃ কয়েকটি অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ হল ঃ

স্মলটক (Small Talk)
টার্বো প্যাসকাল (Turbo pascal)
সি + + (C+ +)
জাভা (Java)
হাইপার টক (Hyper Talk)
ভিজুয়াল বেসিক (Visual Basic)
ভিজুয়াল সি + + (Visual C + +) ইত্যাদি।

৭। অবজেক্ট বলতে কী বোঝায়?

উত্তরঃ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর ক্ষেত্রে ক্লাসে ব্যবহৃত ভেরিয়েবলসমূহের প্রত্যেকটিকে এক একটি অবজেক্ট বলে। প্রতিটি অবজেক্টে কিছু ডাটা থাকে এবং ঐ ডাটা এক্সেস করার জন্য কিছু ফাংশন থাকে।

৮। ক্লাস কী?

উত্তরঃ ক্লাস (Class) হল স্ট্রাকচারের বর্ধিত রূপ এবং স্ট্রাকচারের মতই এক ধরনের ইউজার ডিফাইন্ড ডাটা টাইপ। এটি অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর এমন এক ধরনের মেকানিজম যা স্ট্রাকচারে বিদ্যমান সীমাবদ্ধতা (ডাটার গোপনীয়তা ও বিশ্বস্ততা হ্রাস) দূর করে সিস্টেমে ব্যবহৃত ডাটা মেম্বারকে প্রাইভেট ঘোষণার মাধ্যমে বাইরের অন্য কোন প্রোগ্রাম বা ফাংশনের হস্তক্ষেপ রক্ষা করে ও সিস্টেমের নিরাপত্তা জোরদার করে। প্রতিটি ক্লাসেই কিছু ভেরিয়েবল এবং ঐ ভেরিয়েবলসমূহকে এক্সেস করার জন্য কিছু ফাংশন থাকে।

৯। ইনহেরিটেন্স কাকে বলে?

উত্তরঃ অবজেক্ট অরিয়েন্টেড পদ্ধতিতে একটি ক্লাস প্রয়োজনে অন্য কোন ক্লাসকে ইনহেরিট করার প্রক্রিয়াকে ইনহেরিটেন্স বলে।

১০। ডাটা অ্যাবস্ট্রাকশন কী?

উত্তরঃ ডাটা অ্যাবস্ট্রাকশন হচ্ছে এমন এক ধরনের ডাটা টাইপ বর্ণনা কৌশল যেখানে কিছু ডাটা থাকে এবং ঐ ডাটা এক্সেস করার জন্য কতগুলো ফাংশন থাকে। তবে ঐ টাইপের কোন ভেরিয়েবলের ডাটা তার নিজস্ব ফাংশনের সাহায্য ছাড়া অন্য কোনভাবে এক্সেস করা যায় না।

১১। এনক্যাপসুলেশন বলতে কী বোঝায়?

উত্তরঃ এনক্যাপসুলেশন হচ্ছে এমন একটি প্রক্রিয়া যার মাধ্যমে কোন অবজেক্টের ডাটা ও ফাংশনসমূহের বাইরের অবাঞ্ছিত এক্সেস হতে রক্ষা করা যায় এবং ডাটা ও ফাংশনসমূহকে একত্রিত করে ক্লাস তৈরি করা যায়। এতে অবজেক্টের ডাটা ও ফাংশনসমূহ একটি একক ইউনিটের মত কাজ করে।

১২। পলিমরফিজম কী?

উত্তরঃ পলিমরফিজম শব্দটি 'পলি' ও 'মরফিজম' নামক দুটো গ্রীক শব্দের সমন্বয়ে গঠিত। যাদের অর্থ যথাক্রমে 'বহু' এবং 'রূপ'। অর্থাৎ ভিন্ন ভিন্ন অবস্থায় একই অবজেক্টের 'বহু রূপ' ধারণ করা বা ভিন্ন ভিন্ন ফল প্রদান করার প্রক্রিয়াকে পলিমরফিজম বলে।

১৩। ম্যাসেজ পাসিং বলতে কী বোঝায়?

উত্তরঃ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর ক্ষেত্রে যে প্রক্রিয়ায় অবজেক্টসমূহ পরস্পরের প্রয়োজনে একে অপরের কাছে অনুরোধ পাঠায় এবং প্রেরিত অনুরোধের প্রোক্ষিতে ফলাফল প্রেরণ করে নিজেদের মধ্যে সংযোগ তৈরি করে তাকে ম্যাসেজ পাসিং বলে।

১৪। ডাইনামিক বাইন্ডিং কাকে বলে?

উত্তরঃ যে প্রক্রিয়ায় ফাংশনগুলোর ডাইনামিক রেফারেন্সের ভিত্তিতে প্রোগ্রাম নির্বাহের সময় উপযুক্ত ক্লাসের অবজেক্ট নির্বাচনের মাধ্যমে সঠিক ফাংশন নির্বাচন করা হয় কিংবা পয়েন্টার এক ফাংশন থেকে অন্য ফাংশনে স্থানান্তর করা হয় তাকে ডাইনামিক বাইন্ডিং বলে।

১৫। সি ++ কে হাইব্রিড প্রোগ্রামিং ল্যাংগুয়েজ বলা হয় কেন?

উত্তরঃ সি++, সি ভাষার মত একদিকে যেমন প্রসিডিউর প্রোগ্রামিং এর সুবিধা প্রদান করে এবং একই সাথে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং পদ্ধতির ব্যবহার ও অনুমোদন করে। তাই সি++ কে হাইব্রিড প্রোগ্রামিং ল্যাংগুয়েজ বলে।

১৬। প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং এ কিভাবে সমস্যার সমাধান করা হয়।

উত্তরঃ এই প্রোগ্রামিং পদ্ধতিতে একটি বড় প্রোগ্রামকে বা সমস্যাকে কতগুলো ছোট ছোট অংশ বা মডিউলে ভাগ করা হয়। সমস্যা সমাধানের উপর ভিত্তি করে প্রতিটি অংশের জন্য আলাদা আলাদা ফাংশন ব্যবহার করা হয় এবং পরবর্তীতে সকল ফাংশন সমূহকে একত্রিত করে সমস্যার সমাধান করা হয়।

১৭। প্রসিডিউর অরিয়েন্টেড ল্যাংগুয়েজের অন্যতম প্রধান সমস্যা কি?

উত্তরঃ প্রসিডিউর অরিয়েন্টেড ল্যাংগুয়েজের অন্যতম প্রধান সমস্যা হচ্ছে এতে ব্যবহৃত ফাংশনসমূহ action oriented যা বাস্তব জীবনের সমস্যা সমাধানের সাথে খুব একটা সম্পর্কিত নয়।

১৮। Small Talk কখন এবং কারা আবিস্কার করেন?

উত্তরঃ ১৯৭০ সালে ক্যালিফোর্নিয়ার "Xerox Palo Alto Research Center" এর কম্পিউটার বিজ্ঞানী "Alan Kay" Small Talk আবিষ্কার করেন।

১৯। Turbo Pascal কবে এবং কে ডিজাইন করেন?

উত্তরঃ ১৯৮৪ সালে 'Borland International' এর "Philippe Kahn". Turbo Pascal ডিজাইন করেন।

২০। Hyper Card কী?

উত্তরঃ Hyper card হচ্ছে এমন এক ধরনের electronic card বা card সমষ্টি যাতে প্রয়োজনে data বা information store করা যায়।

সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। সফটওয়্যারের পর্যায়ক্রমিক বিবর্তন উল্লেখ কর।

উত্তর সংকেত ঃ ৩.১ নং অনুচ্ছেদ দ্রষ্টব্য।

২। প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং ও অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর ডাটা ও ফাংশনের মাঝে সম্পর্কের তুলনামূলক চিত্র অঙ্কন কর।

উত্তর সংকেত ঃ ৩.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং এর বৈশিষ্ট্য লেখ।

উত্তর সংকেত ঃ ৩.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর বৈশিষ্ট্য উল্লেখ কর।

উত্তর সংকেত ঃ ৩.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। প্রসিডিউর অরিয়েন্টেড ও অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর মধ্যে পার্থক্য লেখ।

উত্তর সংকেত ঃ ৩.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। ট্রাডিশনাল প্রোগ্রামিং-এর অসুবিধাসমূহ লেখ।

উত্তর সংকেত ঃ ৩.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৭। **উদাহরণসহ ক্লাস ঘোষণার ফরম্যাট বা সিনটেক্স লেখ।**

উত্তর সংকেত ঃ ৩.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৮। **উদাহরণসহ পলিমরফিজম প্রক্রিয়া বর্ণনা কর।**

উত্তর সংকেত ঃ ৩.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৯। **ম্যাসেজ পাসিং প্রক্রিয়া চিত্রসহ বর্ণনা কর।**

উত্তর সংকেত ঃ ৩.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

১০। **অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর সুবিধাসমূহ উল্লেখ কর।**

উত্তর সংকেত ঃ ৩.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১১। **অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর অসুবিধাগুলো কী কী?**

উত্তর সংকেত ঃ ৩.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১২। **অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর ব্যবহার বা প্রয়োগ ক্ষেত্র উল্লেখ কর।**

উত্তর সংকেত ঃ ৩. ৬ নং অনুচ্ছেদ দ্রষ্টব্য।

➥ *রচনামূলক প্রশ্নোত্তর ঃ*

১। **চিত্রসহ সফটওয়্যার বিবর্তনের বিভিন্ন ধাপসমূহ সংক্ষেপে বর্ণনা কর।**

উত্তর সংকেত ঃ ৩.১ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **বিভিন্ন প্রকার অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজের বর্ণনা লেখ।**

উত্তর সংকেত ঃ ৩:৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর উল্লেখযোগ্য বৈশিষ্ট্যসমূহ সংক্ষেপে আলোচনা কর।**

উত্তর সংকেত ঃ ৩.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। **অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর সুবিধা-অসুবিধা আলোচনা কর।**

উত্তর সংকেত ঃ ৩.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

সি++ প্রোগ্রামিং
Overview of C++

অধ্যায় ০৪

৪.০ ভূমিকা ও ইতিহাস (Introduction & History of C++) ঃ

আশি ও নব্বইয়ের দশকে বেশির ভাগ সফটওয়্যার ডেভেলপাররা যে ল্যাংগুয়েজটা বেশি ব্যবহার করতেন তা হল সি। সি বের হওয়ার পর থেকে এখন পর্যন্ত Apple কিংবা PC তে Mac OS, DOS, Windows, Linux প্রভৃতি অপারেটিং সিস্টেম থেকে আরম্ভ করে এসব অপারেটিং সিস্টেমে চালানোর উপযোগী বেশিরভাগ প্রোগ্রাম লেখা হয় সি দিয়ে। সি দিয়ে যতগুলো বড় বড় প্রোগ্রাম লেখা হয়েছে আর কোন প্রোগ্রামিং ল্যাংগুয়েজে তা হয়নি। Embeded system থেকে আরম্ভ করে অপারেটিং সিস্টেম, গ্রাফিক্স এপ্লিকেশন, Client/server ডাটাবেজ, থ্রি ডি গেমস, ডাটা কম্প্রেশন, নেটওয়ার্কিং, ভাইরাস, এন্টি ভাইরাস ইত্যাদি বিভিন্ন ধরনের প্রোগ্রাম লেখা হয় সি দিয়ে। শুধু ডেস্কটপ কম্পিউটারের জন্যই নয়, মেইন ফ্রেমেও প্রোগ্রাম লেখার জন্য সি ল্যাংগুয়েজ ব্যবহার করা হয়। কিন্তু সি দিয়ে এত কিছু করা গেলেও বড় বড় প্রোগ্রাম ডেভেলপ করতে গিয়ে সি এর কিছু সমস্যা ধরা পড়ে। যেমন- কোন প্রোগ্রাম যদি বেশি বড় হয়ে যায় (যেমন 1,00,000 লাইনের হলে) তাহলে সেই প্রোগ্রামে কিছু পরিবর্তন করতে গেলে কিংবা নতুন কিছু যোগ করতে গেলে অনাকাংখিত কিছু ভুল প্রোগ্রামে যোগ হতে পারে। কেননা সিতে প্রোগ্রাম লেখার জন্য Structured বা Procedural Programming পদ্ধতি অনুসরণ করা হয়। আর এই পদ্ধতিতে যেভাবে Data abstraction করা হয়, সেক্ষেত্রে প্রোগ্রাম বড় হয়ে গেলে সব ডাটার প্রতি ঠিকমত নজর দেয়া যায় না। ১৯৮০ সালে Bell Laboratories এর একটা প্রজেক্টের কাজ করতে গিয়ে Bjarne stroustrup এ ধরনের কিছু সমস্যার সম্মুখীন হন। তখন তিনি নতুন একটা প্রোগ্রামিং ল্যাংগুয়েজ ডেভেলপ করার চেষ্টা করেন। কিন্তু এই ল্যাংগুয়েজটা ডেভেলাপ করার সময় তিনি প্রোগ্রামিং ল্যাংগুয়েজের প্রাথমিক উপাদানগুলোর জন্য নতুন syntax তৈরি না করে সে সময়কার সবচেয়ে বেশি ব্যবহৃত ল্যাংগুয়েজ সিকে বেছে নেন এবং সেই সাথে আরো কিছু ল্যাংগুয়েজের ভালো ফিচার এই নতুন ল্যাংগুয়েজে যোগ করেন। যেমনঃ Simula 67 থেকে নেন Object Oriented Programming এর ধারণা, ADA 'র generic এর ধারণাটা যোগ করেন Template হিসাবে ইত্যাদি। আর এভাবে নতুন যে প্রোগ্রামিং ল্যাংগুয়েজ ডেভেলাপ করা হয় তার নাম হল C with Classes.

১৯৮৩ সাল পর্যন্ত নতুন এই ল্যাংগুয়েজটা শুধুমাত্র AT&T র ভেতরই ব্যবহার করা হত। কিন্তু ১৯৮৩ সালের জুলাই মাসে এটাকে প্রথম AT&T 'র বাইরে প্রোগ্রাম ডেভেলাপ করার জন্য ব্যবহার করা হয়। এ বছরই C with Class নামটা Rick Mascitti পরিবর্তন করে রাখেন C++. এই নামকরণের পেছনে যুক্তি হল এরকম, C তে++হল increment অপারেটর এবং C with Classes ল্যাংগুয়েজটা হল C এরই increment ভার্সন। তাই নাম দেয়া হল C++.

C++ এর একটা স্টান্ডার্ড তৈরির জন্য ১৯৮৯ সালের ডিসেম্বর মাসে Hewlett-Packard এর উদ্যোগে X3J16 নামে ANSI একটা কমিটি গঠন করে। X3J16 কমিটি যে স্ট্যান্ডার্ড তৈরি করে ১৯৯১ সালের জুন মাসে তা ISO' র C++ স্ট্যান্ডার্ডের সাথে যুক্ত করা হয়। ১৯৯৫ সালে এই স্ট্যান্ডার্ড সবার জন্য উন্মুক্ত করে দেয়া হয়। বর্তমানে C++ এর যে ANSI স্ট্যান্ডার্ড নির্ধারণ করা হয়েছে সেই স্ট্যান্ডার্ড মেনে যে কোড লেখা হবে তা মাইক্রোসফটের কম্পাইলার দিয়ে যেমন কম্পাইল করা যাবে তেমনি কোন পরিবর্তন ছাড়া Borland- এর কিংবা অন্য কোম্পানির কম্পাইলার দিয়েও কম্পাইল করা যাবে। এছাড়া স্ট্যান্ডার্ড লাইব্রেরীর বিভিন্ন ফাংশন ব্যবহার করে লেখা একই কোড Mac OS, DOS কিংবা Linux এ-ও কম্পাইল করে চালানো যাবে।

বর্তমানে বেশিরভাগ অ্যাপ্লিকেশন ডেভেলাপার ল্যাংগুয়েজ হিসাবে C++ ব্যবহার করেন। যেমনঃ Apple এর Mac OS এবং PC 'র Windows এর ইউজার ইন্টারফেসের বেশিরভাগ অংশ C++ দিয়ে লেখা। এছাড়া Unix বা Linux এবং Windows এর অধিকাংশ লাইব্রেরী C++ এ লেখা।

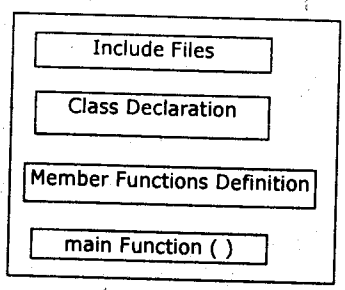
৪.১ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং হিসাবে সি++ এর বৈশিষ্ট্যসমূহ (Features of C++ as OOP) ঃ

সি++ (C++) হচ্ছে বর্তমানকালে বাস্তবভিত্তিক সমস্যা সমাধানে বহুল ব্যবহৃত, আকর্ষণীয় ও জনপ্রিয় অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর যেসব বৈশিষ্ট্যের জন্য প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং এর সাপেক্ষে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর এত জয়জয়কার সেসব বৈশিষ্ট্যের প্রায় সিংহভাগই সি++ প্রোগ্রামিং ল্যাংগুয়েজের মধ্যে বিদ্যমান। তাই সি++ ব্যবহার করে আজ কম্পাইলার, ডাটাবেজ সিস্টেম, কমিউনিকেশন সিস্টেম, রিয়েল টাইম সিস্টেম ডিজাইন ও ডেভেলপ এবং অত্যাধুনিক প্রায় সব ধরনের অ্যাপ্লিকেশন প্রোগ্রাম তৈরি করা সম্ভব হচ্ছে। তবে সি++ প্রোগ্রামিং ল্যাংগুয়েজের এরকম বহুমুখী ব্যবহার ও কার্যকারিতার পেছনে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর বেশকিছু বৈশিষ্ট্য সি++ কে এরূপ অনন্য সাধারণ মর্যাদায় অভিষিক্ত করেছে। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং হিসাবে সি++ (C++) এ যেসব বৈশিষ্ট্য বিদ্যমান সেগুলো হচ্ছে ঃ

- ক্লাস (Class)
- অবজেক্ট (Object)
- ইনহেরিটেন্স (Inheritance)
- ডাটা অ্যাবস্ট্রাকশন (Data abstraction)
- ডাটা হাইডিং (Data hiding)
- এনক্যাপসুলেশন (Encapsulation)
- পলিমরফিজম (Polymorphism)
- ডাইনামিক বাইন্ডিং (Dynamic binding)
- ম্যাসেজ পাসিং (Message passing)
- ফাংশন ওভারলোডিং (Function overloading)
- অপারেটর ওভারলোডিং (Operator overloading)
- গ্যারবেজ কালেকশন (Garbage collection) ও
- অবজেক্ট লাইব্রেরি (Object library) ইত্যাদি।

৪.২ সি++ প্রোগ্রাম স্ট্রাকচার (Structure of C++ program) ঃ

সি++ প্রোগ্রাম এর স্ট্রাকচার নিম্নরূপ ঃ



চিত্র ঃ সি++ প্রোগ্রাম স্ট্রাকচার

সাধারণভাবে প্রতিটি সি++ প্রোগ্রামে উপরোক্ত চারটি অংশ থাকে। অংশগুলো হচ্ছে ঃ

- Include files
- Class declaration
- Member function definitions
- main function ( ).

তবে শুধুমাত্র Include files ও main function ( ) ব্যবহার করেও সাধারণ মানের কিছু প্রোগ্রাম তৈরি করা যায়।

এখানে 'Include files' অংশে প্রয়োজনীয় হেডার ফাইলসমূহ (<iostream.h>, <conio.h> ইত্যাদি) ডিক্লেয়ার করতে হয়। 'Class declaration' অংশের মাধ্যমে প্রোগ্রামের জন্য প্রয়োজনীয় এক বা একাধিক ক্লাস (Class) ঘোষণা করা হয়। প্রতিটি ক্লাসের ভিতরে 'private', 'public' ও protected' অংশে প্রয়োজনীয় ভেরিয়েবল ও ফাংশন ঘোষণা করা হয়। 'Member function definitions' অংশে প্রোগ্রামে ব্যবহৃত ক্লাস বা ক্লাসসমূহে ডিক্লেয়ারকৃত ফাংশনসমূহকে ডিফাইন (Define) করা হয়। 'main Function' অংশে প্রতিটি ক্লাসের অবজেক্ট (Object) ঘোষণার মাধ্যমে সংশ্লিষ্ট অবজেক্টের মাধ্যমে সংশ্লিষ্ট ক্লাসের ফাংশনকে কল (Call) করা হয়। তাছাড়া main ( ) ফাংশন অংশে অন্যান্য লাইব্রেরি ফাংশন ও ইউজার ডিফাইন্ড ফাংশনসমূহ ও কল (Call) করা হয়। যেমন- clrscr ( ) ফাংশন, getch ( ) ফাংশন ইত্যাদি। উল্লেখ্য যে, সি (C) ভাষার ন্যায় সি++ ভাষায় ও প্রতিটি প্রোগ্রামে একটি main ( ) ফাংশন থাকে এবং main ( ) ফাংশন থেকেই প্রোগ্রাম নির্বাহ শুরু হয়।

ফরম্যাট ঃ

নিম্নে একটি সি++ (C++) প্রোগ্রামের ফরম্যাট দেখানো হল ঃ

Include Files

```
# include < iostream.h>
# include <conio.h>
```

Class declaration

```
class  add
{ private :
        int number1, number2, result;
  public :
        void sum ();
} ;
```

Member Function Definition

```
void add : : sum ( )
{
cout << "Enter the value of number1 and number2 :"
cin >> number1 >> number2;
result = number1 + number2;
cout <<"Summation is =" <<result;
}
```

main ( ) ফাংশন

```
void main ( )
{ clrscr ( );
add M; // object declaration
M. sum ( ); // function calling
getch ( );
}
```

৪.৩ সি++ ইনপুট-আউটপুট স্ট্রিম ও সি++ ক্লাস স্ট্রিম (C++ I/O stream and C++ class stream) ঃ

সি ভাষায় ইনপুট-আউটপুট অপারেশনের জন্য একটি শক্তিশালী ইনপুট-আউটপুট ফাংশন সেট scanf ( ) ও printf ( ) আছে। সি++, সি এর সুপারসেট হওয়ায় সি এর সকল ইনপুট-আউটপুট ফাংশন এতে ব্যবহার করার কথা। কিন্তু দুইটি প্রধান সমস্যার কারণে সি++ ভাষায় সি এর ইনপুট-আউটপুট ফাংশনগুলো ব্যবহার করা যায় না।

– **প্রথমত,** সি এর আই/ও ফাংশনসমূহ কেবল বিল্ট-ইন ডাটা টাইপ ভেরিয়েবলের জন্য প্রযোজ্য, তাই এগুলো কোন অবজেক্ট নিয়ে কাজ করতে পারে না।

– **দ্বিতীয়ত,** সি++ এ অবজেক্ট ছাড়াও আরো কতগুলো জটিল বিষয় যেমন, বেস ক্লাস, ডিরাইভড ক্লাস, ভারচুয়াল ফাংশন, অপারেটর ওভারলোডিং, ফাংশন ওভারলোডিং, পলিমরফিজম ইত্যাদি নিয়ে কাজ করতে হয়।

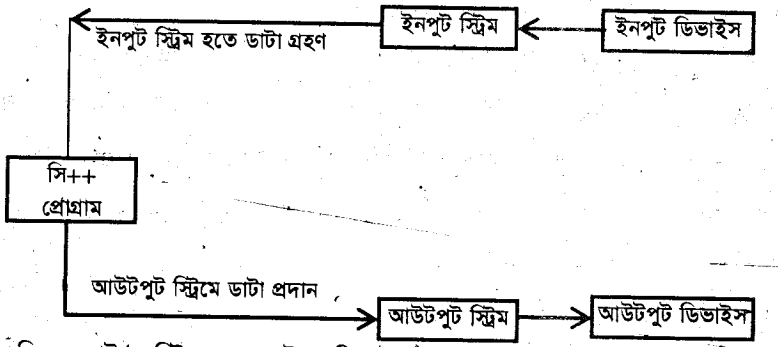
এসব সমস্যা সমাধানের জন্য সি++ এর একটি নিজস্ব আই/ও সিস্টেম (ফাংশন এবং অপারেটর সেট) আছে যা অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং পদ্ধতির বিভিন্ন বৈশিষ্ট্যসমূহ অনুমোদন করে। সি++ আই/ও সিস্টেম ক্লাস, অবজেক্ট, বেস ক্লাস, ডিরাইভড ক্লাস, ভারচুয়াল ফাংশন, অপারেটর ওভারলোডিং, ফাংশন ওভারলোডিং, পলিমরফিজম ইত্যাদি জটিল বিষয়সমূহ নিয়ে বিভিন্ন অপারেশন করতে সক্ষম।

যে কোন আই/ও অপারেশনের জন্য প্রোগ্রাম এবং কম্পিউটারের বিভিন্ন আই/ও ডিভাইসের (যেমন, কীবোর্ড, মনিটর, ডিস্ক ড্রাইভ, টেপ ড্রাইভ, প্রিন্টার, টারমিনাল ইত্যাদি) মধ্যে নিবিড় সংযোগ থাকা আবশ্যক। প্রোগ্রাম এবং এসব আই/ও ডিভাইসের মধ্যে সংযোগ স্থাপনের মূলে রয়েছে এক বিশেষ ধরনের স্ট্রিম অবজেক্ট। স্ট্রিমসমূহ এক গুচ্ছ বাইট ছাড়া আর কিছু নয়। স্ট্রিম দুই রকমের হতে পারে ঃ

- **আই/ও স্ট্রিম** (I/O stream) এবং
- **ক্লাস স্ট্রিম** (Class stream)।

আই/ও স্ট্রিম হল ইনপুট/আউটপুট সম্পাদনকারী এক গুচ্ছ বাইটের সমষ্টি, আর ক্লাস স্ট্রিম হল পরস্পর সম্পর্কযুক্ত কতকগুলো ক্লাসের সমষ্টি।

সি++ ইনপুট-আউটপুট স্ট্রিম (C++ I/O stream) ঃ আই/ও স্ট্রিমসমূহ প্রোগ্রাম এবং বিভিন্ন আই/ও ডিভাইসের মধ্যে ডাটা আদান-প্রদানের জন্য ইন্টারফেস হিসেবে কাজ করে। সুতরাং স্ট্রিমসমূহকে ডাটা ইনপুট করার জন্য সোর্স স্ট্রিম হিসেবে অথবা ডাটা পাঠানোর জন্য ডেস্টিনেশন স্ট্রিম হিসেবে ব্যবহার করা হয়। প্রোগ্রামে ডাটা সরবরাহ করার জন্য ব্যবহৃত সোর্স স্ট্রিমকে ইনপুট স্ট্রিম এবং প্রোগ্রাম হতে ডাটা পাঠানোর জন্য ব্যবহৃত ডেস্টিনেশন স্ট্রিমকে আউটপুট স্ট্রিম বলা হয়। অর্থাৎ, প্রোগ্রামে সোর্স স্ট্রিম হতে কীবোর্ড, মাউস কিংবা অন্য কোন ইনপুট ডিভাইস দ্বারা ডাটা গ্রহণ করা হয় এবং প্রয়োজনীয় প্রক্রিয়াকরণ শেষে মনিটর, প্রিন্টার বা অন্য কোন আউটপুট ডিভাইসের মাধ্যমে আউটপুট স্ট্রিমে ডাটা ফেরত পাঠান হয়। চিত্রে আই/ও স্ট্রিম এবং আই/ও ডিভাইসের মধ্যে ডাটা আদান-প্রদানের প্রক্রিয়া দেখান হল।



চিত্র ঃ আই/ও স্ট্রিম এবং আই/ও ডিভাইসের মধ্যে ডাটা আদান-প্রদান প্রক্রিয়া

সি++ কম্পাইলার এরূপ কতগুলো বিল্ট-ইন অবজেক্ট স্ট্রিম আছে যেগুলো প্রোগ্রাম নির্বাহের সময় স্বয়ংক্রিয়ভাবে সক্রিয় হয়। বহুল ব্যবহৃত এরূপ দুইটি অবজেক্ট স্ট্রিম হল ঃ cin এবং cout। প্রোগ্রামে cin ইনপুট স্ট্রিম বা সোর্স স্ট্রিম হিসেবে কাজ করে যা কীবোর্ড হতে ডাটা গ্রহণ করে, আর cout আউটপুট স্ট্রিম বা ডেস্টিনেশন স্ট্রিম হিসেবে কাজ করে যা মনিটরে কোন ডাটা ভ্যালু প্রদর্শন করে।

cin স্টেটমেন্ট (cin statement) ঃ cin স্টেটমেন্টের ফরম্যাট নিম্নরূপ ঃ

```
cin>> expression;
```

অথবা

```
cin>> expression1>> expression2>> expresson3;
```

এখানে 'cin' সি++ এর ios ক্লাসের এক বিশেষ ধরনের প্রিডিফাইন্ড স্ট্রিম অবজেক্ট যা প্রোগ্রাম নির্বাহের সময় কম্পিউটারের CPU এর সাথে স্ট্যান্ডার্ড ইনপুট ডিভাইসের (যেমন- কীবোর্ড) সংযোগ স্থাপন করে। '>>' অপারেটরটিকে ইনপুট অপারেটর বা এক্সট্রাকশন (extraction) অপারেটরও বলা হয় যা ইউজার কর্তৃক কী বোর্ডের মাধ্যমে কোন ভেরিয়েবলের মান গ্রহণ করে এবং 'expression' বা 'expresion1' 'expression2' ইত্যাদি হল এক বা একাধিক ভেরিয়েবলের নাম।

উদাহরণ ঃ

```
cin>> number;
```

অথবা,

```
cin>> length>>width;
```

উপরোক্ত স্টেটসমূহের মাধ্যমে 'number' নামক একটি ভেরিয়েবলের মান অথবা 'length' ও 'width' নামক দুটি ভেরিয়েবলের মান ইনপুট নেয়া হয়।

cout স্টেটমেন্ট (cout statement) ঃ cout স্টেটমেন্ট ফরমাট নিম্নরূপ ঃ

```
cout << expression;
```

অথবা

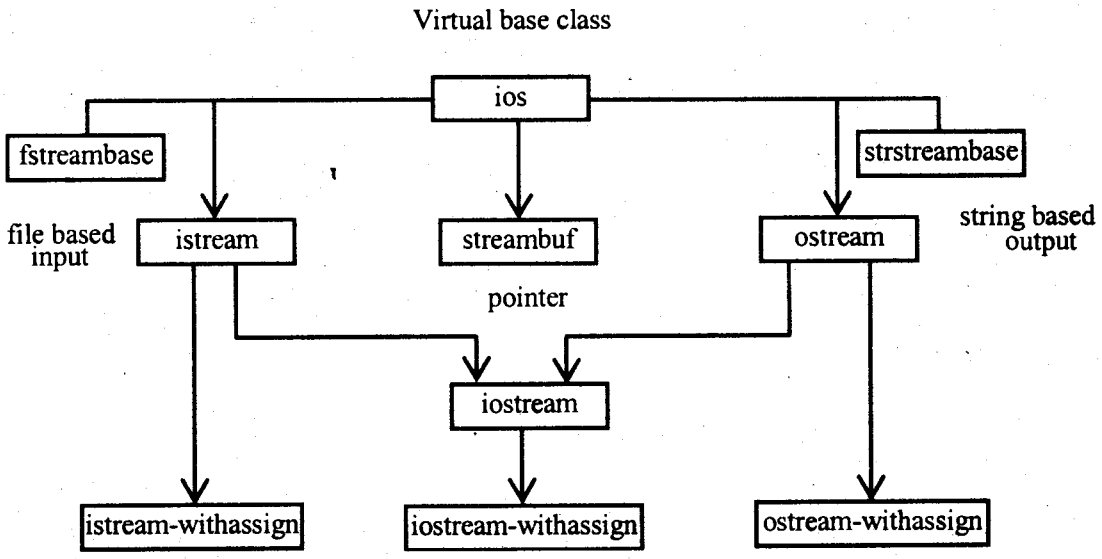
```
cout << expression1 << expression2;
```

এখানে, 'cout' সি++ এর ios ক্লাসের এক বিশেষ ধরনের প্রিডিফাইন্ড স্ট্রিম অবজেক্ট বা প্রোগ্রাম নির্বাহের সময়ে কম্পিউটারের সিপিইউ এর সাথে স্ট্যান্ডার্ড আউটপুট ডিভাইসের (যেমন মনিটর) সংযোগ স্থাপন করে ও Expression এর মান মনিটরে প্রদর্শন করে। '<<' হল আউটপুট অপারেটর বা ইনসার্শন (Insertion)-অপারেটর যার মাধ্যমে কোন আউটপুট ভেরিয়েবলের মান মনিটরে প্রদর্শিত হয়। অপরদিকে 'expression' বা 'expression1', 'expression2' হল আউটপুট ভেরিয়েবলের নাম।

উদাহরণ ঃ cout << result;

উপরোক্ত স্টেটমেন্টের মাধ্যমে 'result' নামক একটি আউটপুট ভেরিয়েবলের মান মনিটরে প্রদর্শিত হবে।

সি++ ক্লাস স্ট্রিম (C++ class stream) ঃ



চিত্র ঃ ক্লাস স্ট্রিম

বেসিক ইনপুট আউটপুট অপারেশন সম্পন্ন করার জন্য সি++ এর আই/ও সিস্টেম একগুচ্ছ হায়ারারকিক্যাল ক্লাস ধারণ করে। এই ক্লাসসমূহকে ক্লাস স্ট্রিম বলা হয়। ক্লাস স্ট্রিমসমূহের মধ্যে ios, istream, ostream এবং iostream অন্যতম। istream (ইনপুট স্ট্রিম) এবং ostream (আউটপুট স্ট্রিম) ios স্ট্রিম ক্লাসের বেস ক্লাস। istream এবং ostream আবার iostream ক্লাসের বেস ক্লাস। ios ক্লাসকে ভারচুয়াল বেস ক্লাস রূপে বর্ণনা করা হয়েছে যেন ios ক্লাসের মেম্বারসমূহ iostream ক্লাসে একবারের বেশি ইনহেরিট না হয়। চিত্রে স্ট্রিম ক্লাসসমূহের ইনহেরিটেন্স রিলেশনশীপ দেখান হয়েছে। এসব ক্লাসের প্রটোটাইপ <iostream.h> নামের হেডার ফাইলে বর্ণনা করা আছে। এজন্য সি++ প্রোগ্রামে কোন আই/ও স্ট্রিম ব্যবহার করলে তার প্রটোটাইপ হিসেবে প্রোগ্রামে <iostream.h> হেডার ফাইল সংযুক্ত করতে হয়।

ios ক্লাস সকল প্রকার বেসিক ফরম্যাটেড এবং আনফরম্যাটেড ইনপুট/আউটপুট অপারেশন অনুমোদন করে, istream ক্লাস সকল প্রকার বেসিক ফরম্যাটেড এবং আনফরম্যাটেড ইনপুট অপারেশন অনুমোদন করে এবং ostream ক্লাস সকল প্রকার বেসিক ফরম্যাটেড এবং আনফরম্যাটেড আউটপুট অপারেশন অনুমোদন করে। iostream ক্লাস সকল প্রকার ইনপুট/আউটপুট স্ট্রিম অপারেশনের জন্য ব্যবহৃত হয়। istream-withassign, ostream-withassign এবং iostream-withassign ক্লাসসমূহ ios, istream, ostream এবং iostream ক্লাসসমূহে এসাইনমেন্ট অপারেটর যুক্ত করে।

প্রোগ্রামিং ল্যাংগুয়েজ-২৫০

নিম্নে সি++ ক্লাস স্ট্রিমসমূহের বৈশিষ্ট্যসমূহ টেবিল আকারে দেখানো হল।

ক্লাস	বৈশিষ্ট্য
ios (কমন ইনপুট/আউটপুট স্ট্রিম)	সকল প্রকার বেসিক ইনপুট/আউটপুট অপারেশনের সুবিধাবলি প্রদান করে।
	ফরম্যাটেড ইনপুট/আউটপুট অপারেশনে ব্যবহৃত ফাংশনসমূহের ডেসক্রিপশন (বর্ণনা) ধারণ করে।
	streambuf নামক বাফার অবজেক্টের একটি পয়েন্টার ধারণ করে।
istream (ইনপুট স্ট্রিম)	ios ক্লাসের সকল বৈশিষ্ট্য ধারণ করে।
	ওভারলোডেড এক্সট্রাকশন অপারেটর (>>) ধারণ করে।
	ফরম্যাটেড এবং আনফরম্যাটেড ইনপুট ফাংশনসমূহের (যেমন get ( ), getline ( ), read ( )) সুবিধাবলী প্রদান করে।
ostream (আউটপুট স্ট্রিম)	ios ক্লাসের সকল বৈশিষ্ট্য ধারণ করে।
	ওভারলোডেড ইনসারশন অপারেটর (<<) ধারণ করে।
	ফরম্যাটেড এবং আনফরম্যাটেড আউটপুট ফাংশনসমূহের (যেমন, put ( ), write ( )) সুবিধাবলী প্রদান করে।
iostream (ইনপুট/আউটপুট স্ট্রিম)	মাল্টিপল ইনহেরিটেন্স পদ্ধতিতে ios, istream এবং ostream ক্লাসের বৈশিষ্ট্য অর্জন করে। সুতরাং সকল ইনপুট এবং আউটপুট অপারেশনে ব্যবহৃত ফাংশনসমূহের ডেসক্রিপশন (বর্ণনা) ধারণ করে।
streambuf (স্ট্রিম বাফার পয়েন্টার)	স্ট্রিম বাফারের মাধ্যমে ইনপুট/আউটপুট স্ট্রিম এবং ফিজিক্যাল ডিভাইসসমূহের মধ্যে সংযোগ (ইন্টারফেস) স্থাপন করে।
	ফাইল অপারেশনে ব্যবহৃত filebuf ক্লাসের বেস ক্লাস হিসেবে কাজ করে।

টেবিল ঃ সি++ এর ক্লাস স্ট্রিমসমূহের বৈশিষ্ট্য

৪.৩.১ সি++ ইনপুট-আউটপুট ফরম্যাট (C++ Input-Output format) ঃ

সি++ ভাষায় cin ও cout স্টেটমেন্ট ব্যবহারে বেশকিছু সীমাবদ্ধতা আছে। যেমন cin ব্যবহার করে ফাঁকাস্থান বা ট্যাব বিশিষ্ট কোন মান ইনপুট করা যায় না বা ফাঁকা স্থান থাকলেও ফাঁকাস্থান বা ট্যাবের পরবর্তী অংশ ইনপুট হতে বাদ যায় কিংবা পরবর্তী ভেরিয়েবলের মান হিসেবে গৃহীত হয়। আবার cout দ্বারা ভগ্নাংশ বিশিষ্ট কোন সংখ্যার সঠিক মান প্রদর্শিত না হয়ে কাছাকাছি মান প্রদর্শিত হয়।

এরূপ সীমাবদ্ধতা দূর করার জন্য কতগুলো মেম্বার ফাংশন, কন্ট্রোল ফ্লাগ এবং ম্যানিপুলেটর ব্যবহৃত হয়। এসব ইনপুট আউটপুট অপারেশনসমূহকে দুইভাগে ভাগ করা যায়। যথা ঃ

- আনফরম্যাটেড ইনপুট-আউটপুট অপারেশন ও
- ফরম্যাটেড ইনপুট-আউটপুট অপারেশন।

আনফরম্যাটেড ইনপুট-আউটপুট অপারেশন ঃ আনফরম্যাটেড ইনপুট-আউটপুট অপারেশনের জন্য ব্যবহৃত ফাংশনগুলোর মধ্যে get ( ), put ( ), getline ( ) ও write ( ) ফাংশন অন্যতম।

get ( ) ও put ( ) ফাংশন ঃ কোন character ইনপুট নেয়ার জন্য get ( ) ফাংশন এবং গ্রহণকৃত character মনিটরে তথা আউটপুটে প্রদর্শনের জন্য put ( ) ফাংশন ব্যবহার করা হয়।

get ( ) এবং put ( ) ফাংশন কল করার জন্য cin এবং cout অবজেক্টের সাথে ডট্ (.) অপারেটর ব্যবহার করা হয়। যথা –

```
char ch;
cin.get (ch);
cout.put(ch);
```

নিম্নে get ( ) এবং put ( ) ফাংশন ব্যবহার করে একটি উদাহরণ দেখা যাক–

```
# include <iostream.h>
void main ( )
{
int count = 0;
char ch;
cout <<" Enter a line of Text : " << endl;
ch = cin.get ( );
while (ch! = '\n')
{
cout.put (ch);
count ++;
ch = cin.get ( );
}
cout <<" \n Characters (including White Spaces) : " ;
cout << count;
}
```

আউটপুট ঃ

```
Enter a line of Test :
Diploma in Computer Engineering
Characters (Including White Spaces) : 31
```

getline ( ) ও write ( ) : সি++ প্রোগ্রামে কোন স্ট্রিং এর character সমূহ get ( ) ফাংশন ব্যবহার করে একটি একটি করে ইনপুট গ্রহণ করার পরিবর্তে একত্রে ইনপুট করার জন্য getline ( ) ফাংশন ব্যবহার করা যায়।

একইভাবে, put ( ) ফাংশন ব্যবহার করে একটি একটি character আউটপুট প্রদানের পরিবর্তে একত্রে আউটপুট প্রদান করার জন্য write ( ) ফাংশন ব্যবহার করা যায়। পূর্বের মত এই getline ( ) এবং write ( ) ফাংশন ব্যবহার করে কার্যনির্বাহের জন্য cin ও cout অবজেক্টের সাথে ড্ট (.) অপারেটর ব্যবহার করে নিতে হয়। getline ( ) ও write ( ) ফাংশনের ফরম্যাট হল–

```
cin.getline (line, size);
cout.write (line, size);
```

এখানে line হচ্ছে ফাংশনের ব্যবহৃত ভেরিয়েবল এবং size হচ্ছে লাইন ভেরিয়েবলে ব্যবহৃত স্ট্রিং এর সর্বোচ্চ character সংখ্যা। এই দুইটি ফাংশন ব্যবহার করে প্রোগ্রাম দেখলে আশা করি ফাংশনগুলোর ব্যবহার সহজ হয়ে যাবে।

প্রোগ্রাম ঃ

```
# include < iostream.h >
void main ( )
{
char *name, *dpt;
cout <<" Enter Name : ";
cin.getline (name, 14);
cout <<" Enter Subject : ";
cin.getline (dpt, 9);
cout <<" \n Name : " <<name <<endl;
cout <<" Dept : " <<dpt;
}
```

আউটপুট ঃ

```
Enter Name : Mahfuza Rahman Nila
Enter Subject : Computer Engineering
Name : Mahfuza Rahman
Subject : Computer Eng
```

write ( ) ফাংশন ব্যবহার করে প্রোগ্রাম ঃ

```
# include < iostream.h>
void main ( )
{
int size = 50;
char *iine;
cout <<" \n Enter a line of Text : ";
cin.getline (line, size);
cout <<" \n Output (using cout)  : " <<line;
cout <</ "\n Output (using write) : " <<endl;
int L = strlen (line);
for (int i = 1; i <= L; i = i + 2)
{
cout.write (line, i);
cout << endl;
}
}
```

আউটপুট ঃ

```
Enter a line of text : Computer Technology
Output (using cout) : Computer Technology
Output (using write) :
C
Com
Compu
Compute
Computer
Computer Te
Computer Tech
Computer Techno
Computer Technolo
Computer Technology
```

ফরম্যাটেড ইনপুট-আউটপুট অপারেশন ঃ সি++ প্রোগ্রামিং ল্যাংগুয়েজে ফরম্যাটেড ইনপুট আউটপুট অপারেশনের কাজ তিনভাবে করা যায়। যথা ঃ

- ম্যানিপুলেটর (Mainpulator) ফাংশন ব্যবহার করে
- ios ক্লাসের মেম্বার ফাংশন ব্যবহার করে এবং
- ইউজার ডিফাইন্ড ফাংশন ব্যবহার করে।

ম্যানিপুলেটর হল বিশেষ ধরনের স্ট্রিম ফাংশন যা আউটপুট ভ্যালু পরিবর্তন করতে পারে এবং যা ব্যবহার করে সুবিধাজনক ও নির্দিষ্ট ফরম্যাটে ডাটা প্রদর্শন করা যায়। সি++ এ ব্যবহৃত মনিপুলেটর দুই প্রকার। যথা ঃ

- ম্যানিপুলেটর ফাংশন (setw ( ), setfill ( ), setprecession ( ) ইত্যাদি।
- ম্যানিপুলেটর অপারেটর (endl, ends, ws ইত্যাদি।)

মেম্বার ফাংশন	ব্যবহার	সমতুল্য ম্যানিপুলেটর ফাংশন
width ( ) cout.width(5);	আউটপুট ভ্যালু প্রদর্শনের জন্য ফিল্ডের প্রয়োজনীয় দৈর্ঘ্য নির্ধারণ করে।	setw ( ) cout << setw(5);

fi11 ( )
cout.fill('*');
cout.fill(42);
	আউটপুট ফিল্ডে ব্যবহৃত স্থান পূরণের জন্য কোন ক্যারেক্টোর নির্ধারণ করে।	setfill (c)
		cout<<setfill ('*');
		cout<<setfill(42);

precision ( )
cout.precision(2)
cout.precision(5);
	ভগ্নাংশ টাইপ মান প্রদর্শনে দশমিকের পরে কত ডিজিট থাকবে তা নির্ধারণ করে।	setpricision (d)
		cout<<setprecision(2);
		cout<<setprecision(5);

setf ( )
cout.setf(ios::oct, ios ::basefuekd);
cout.setf(ios::left, ios::adjustfield);
cout.setf(ios::hex, ios::basefuekd);
	আউটপুট লেফট ও রাইট জাস্টিফিকেশন, বিভিন্ন সায়েন্টেফিক ফরম্যাটে প্রদর্শনের জন্য ব্যবহৃত কন্ট্রোল ফ্লাগ সেট করে।	setiosflags ( )
		cout<<setiosflags(ios::oct);
		cout<<setiosflags(ios::left);
		cout<<setiosflags(ios::hex);

reset ( ) cout.unsetf(ios::left);	setf ফ্লাগ দ্বারা সেট করা কন্ট্রোল ফ্লাগ রিসেট করে।	ressetiosflags ( ) cout<<resetiosflags (ios::left);

টেবিল ঃ ক্লাসের মেম্বার ফাংশন ও তাদের সমতুল্য ম্যানিপুলেটর ফাংশনের ব্যবহার

width ( ) function : output value প্রদর্শনের জন্য প্রয়োজনীয় field size specify করার জন্য width ( ) ফাংশন ব্যবহৃত হয়।

এর format নিম্নরূপ ঃ

```
cout. width (w);
```

এখানে w হলো field width. width ( ) functionটি কেবলমাত্র একটি item এর field width specify করতে পারে।

prcission ( ) function :

কোন পূর্ন সংখ্যার সাথে দশমিক ও তার পরের কয়েকটি নির্দিষ্ট ঘর (6) পর্যন্ত প্রদর্শন করানোর জন্য এই function টি ব্যবহৃত হয়। এর গঠন নিম্নরূপ ঃ

```
cout. precission (d);
```

এখানে d হচ্ছে দশমিকের পর কতঘর পর্যন্ত সংখ্যা বসবে তার identification নম্বর। উদাহরণ নিম্নরূপ ঃ

```
cout. precission (3);
cout << sqrt (2)
cout << 3.14159
```

আউটপুট ঃ 1.141 & 3.142 (nearest cent by rounded)

fill ( ) function :

অনেক সময় যে value print করতে হবে তার চেয়ে অনেক বেশি field থাকে। ফলে ঐ value গুলো print করার পর বাকি field গুলো খালি থাকে, এই খালি field গুলো white space হিসেবে default থাকে। ইচ্ছে করলে আমরা কোন desired character ব্যবহার করে ঐ field গুলো পূর্ণ করতে পারি। নিম্নে এর function structure দেখানো হল।

```
cout. fill (ch) ;
```

উদাহরণ ঃ

```
cout. fill (@,)
cout. width (7);
cout << "352" << "\n";
```

আউটপুট ঃ

@	@	@	@	3	5	2

অর্থকরী প্রতিষ্ঠান ও ব্যাংকসমূহ এই ধরণের Padding প্রিন্টিং চেকে ব্যবহার করে যেন কেউ সহজে amount change করতে না পারে।

setf ( ) :

width ( ) function এর সাহায্যে text বা number সমূহ কেবলমাত্র right side থেকে print হয় কিন্তু seft() ফাংশনের সাহায্যে left to right কিংবা, right to left অথবা internally text কিংবা number field এ print করা যায়। setf মূলতঃ এক ধরণের flag (set flag) যা ios class এর একটি member function। নিম্নে এর গঠন ও উদাহরণ দেয়া হল।

```
cout. setf (arg1, arg2);
```

উদাহরণ-১।

```
cout. fill ('*');
cout. setf (ios: : left, ios :: adjust field);
cout. width (15);
cout << "Table 1" << "\n";
```

আউটপুট ঃ

T	a	b	l	e		1	*	*	*	*	*	*	*	*

উদাহরণ-২।

```
cout. fill ('*')
cout. precession (4);
cout. setf (ios : : internal, ios : : adjust field);
cout. setf (ios : : scientific, ios : : float field);
cout. width (15);
cout < < -12.34567 << "\n";
```

আউটপুট ঃ

-	*	*	*	*	*	1	.	2	3	5	e	+	0	1

ম্যানিপুলেটরের ব্যবহার ঃ

C ++ এ formatted i/o র আরেকটি উপায় হল manipulator. এই manipulator ব্যবহার করতে হলে প্রোগ্রামে <iomanip. h> header ফাইল include করতে হবে।

```
#include <iostream. h> //1987 standard specification of C++
# include <iomanip.h>
void main ( )
{
bool b = false;
cout << "without manipulator\ "b\" = " <<b<<endl;
cout <<"with manipulator\" b\" = "<< boolalpha << b;
}
```

আউটপুট ঃ

```
Without manipulator  "b" = 0
With manipulator "b" = false.
```

এই প্রোগ্রামটা b ভেরিয়েবলের মান প্রথমে 0 দেখায় এবং পরে cout এর সাথে boolalpha ব্যবহার করার ফলে b এর ডাটাকে boolean মান হিসাবে দেখায়। এখানে boolalpha হল একটা manipulator যা কোন bool টাইপ ভেরিয়েবলের ডাটাকে boolean হিসাবে দেখায়। এভাবে formatted i/o'র জন্য প্রোগ্রামে যেসব manipulator ব্যবহার করা যায় নীচে তাদের তালিকা দেওয়া হল ঃ

manipulator	ব্যবহার
boolalpha	এটা স্ট্রিমকে bool ডাটা টাইপ হিসাবে দেখানোর জন্য ব্যবহার করা হয়।
dec	এটা ডাটাকে numeric টাইপে দেখানোর জন্য ব্যবহার করা হয়।
endl	এটা data stream এ newline ক্যারেকটার যুক্ত করে দেয়। এই manipulator শুধুমাত্র আউটপুটের কাজে ব্যবহার করা যায়।
ends	আউটপুট stream-এর শেষে NULL ক্যারেকটার যোগ করে।
fixed	আউটপুট stream এর fixed flag সেট করে।
flush	আউটপুট stream এর বাফার flush করার জন্য এই manipulator ব্যবহার করা হয়।
hex	সংখ্যাকে হেক্সাডেসিমালে দেখানোর জন্য এই manipulator ব্যবহার করা হয়।
internal	Internal flag সেট করার জন্য এই manipulator ব্যবহার করা হয়।
left	আউটপুটকে left justified করে।
noboolalpha	boolalpha flag অফ করার জন্য এই manipulator ব্যবহার করা হয়।
noshowbase	আউটপুট stream এর showpoint flag অফ করার জন্য ব্যবহার করা হয়।
noshowpoint	আউটপুট stream এর showpoint flag অফ করার জন্য ব্যবহার করা হয়।

টেবিল ঃ বিভিন্ন প্রকার ম্যানিপুলেটরের ব্যবহার

endl :

C++ প্রোগ্রাম এ বহুল ব্যবহৃত endl ম্যানিপুলেটর ক্যারেইজ রিটার্ন বা নিউ লাইন ক্যারেক্টার (' \l ') হিসেবে কাজ করে। নিম্নে endl ম্যানিপুলেটর ব্যবহার করে একটি প্রোগ্রাম লিখা হল।

```
# include < iostream.h>
void main ( )
{
int x = 500, y = 300;
cout << "x = "<<x<<" \n "<<"y = "<<y;
cout << endl;
cout <<"x = "<<x<<endl <<"y = "<<y;
}
```

আউটপুট ঃ

```
x = 500
y = 300
x = 500
y = 300
```

অর্থাৎ cout <<"\n"; এবং cout << endl; স্টেটমেন্ট দুইটি সমতুল্য।

ends :

কোন স্ট্রিং বা সংখ্যার শেষে একটি নাল ক্যারেকটার (' \ o') সংযোগ করার জন্য ends ম্যানিপুলেটর ব্যবহৃত হয়। নিম্নে ends ম্যানিপুলেটর ব্যবহার করে একটি প্রোগ্রাম লিখা হল ঃ

```
# include < iostream. h>
# include < stream. h>
void main ( )
{
char *S1 = "Computer";
char *S2 = "Technology";
cout << S1 << S2 << endl;
cout << S1 << ends <<S2 << endl;
cout << S1 << "  " <<S2 << endl;
cout << S1 << ends << ends << S2;
}
```

আউটপুট ঃ

```
Computer Technology
Computer Technology
Computer Technology
Computer Technology
```

অর্থাৎ cout << " ", এবং cout << ends; স্টেটমেন্ট দুইটি সমতুল্য।

ws :

C++ প্রোগ্রামে কীবোর্ড দ্বারা কোন চলকের মান গ্রহনের সময় প্রথম শব্দ বা ক্যারেক্টারের শুরুতেই যদি কোন ফাঁকা স্থান (White space) পড়ে তবে তা বাদ দেওয়ার জন্য ws ম্যানিপুলেটর ব্যবহৃত হয়।

ws ম্যানিপুলেটর ব্যবহারের ফরম্যাট হল ঃ

```
cin >> ws;
```

নিম্নে ws ম্যানিপুলেটর ব্যবহার করে একটি প্রোগ্রাম লিখা হল।

```
# include < iostream. h>
# include < conio. h>
void main ( )
{
char *Dept;
char *Roll;
cout <<"\nEnter Dept :";
cin >> ws;
cin.getline (Dept, 30);
cout << "Entered Dept :" <<Dept;
}
```

আউটপুট ঃ

```
Enter Dept : Computer Scince & Information Technology
Entered Dept : Computer Science & Information Technology.
```

৪.৪ সি++ এ ব্যবহৃত কীওয়ার্ড, ভেরিয়েবল, কন্সট্যান্ট, অপারেটর ও কন্ট্রোল স্টেটমেন্টসমূহ (Keyword, Variable, Constant, Operator and Control Statements in C++) ঃ

কীওয়ার্ড (Keyword) ঃ

C-তে যেসব keyword আছে C++ এ ও সেই keyword গুলো নিয়ে কাজ করা যায়। তবে এগুলো ছাড়াও অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং-এর সুবিধা প্রদানের জন্য C++ এ আরো কিছু নতুন keyword যুক্ত করা হয়েছে। নতুন এই keyword গুলো সহ ANSI/ISO C++ standard অনুযায়ী C++ এ মোট 63টি keyword আছে। নিম্নে সেগুলোর তালিকা দেয়া হল ঃ-

সি++ কীওয়ার্ড (C++ Keywords)			
asm	default	float	operator
auto	delete	for	private
bool	do	friend	protected
break	double	go to	public
case	dynamic cast	if	register
catch	else	inline	reinterpret–cast
char	enum	int	return
class	explicit	long	short
const	export	mutable	signed
	extern		
const_cast	false	namespace	sizeof
continue	struct	new	static
static_cast	throw	switch	template
		true	
this	typeid	typename	try
typedef	using	virtual	union
unsigned	wchar_t	while	void
			volatile

টেবিল ঃ কীওয়ার্ড (Keywords)

ভেরিয়েবল (Variable) ঃ

সি ভাষার ন্যায় সি++ এ ও Variable ৪ ধরনের। যথা ঃ

- ইন্টিজার টাইপ (int)
- ক্যারেক্টার টাইপ (char)
- ফ্লোটিং পয়েন্ট (float)
- ডাবল টাইপ (double)।

ভেরিয়েবল ঘোষণা (Variable Declaration) ঃ

ভেরিয়েবল ঘোষণার নিয়ম হল ঃ

```
DataType VariableName (s);
```

উদাহরণ ঃ

```
char ch;
int length, width;
float result;
double result;
```

তাছাড়া ভেরিয়েবল নামকরণের নিয়মাবলি ও C ভাষার মতই।

কন্সট্যান্ট (Constant) ঃ সি++ এ ব্যবহৃত কন্সট্যান্ট ও দুই ধরনের। যথা ঃ

- লিটারেল কন্সট্যান্ট (Literal Constant)
- সিম্বলিক কন্সট্যান্ট (Symbolic Constant)

এছাড়া সিম্বলিক কন্সট্যান্ট আবার নানা রকম হতে পারে। যেমন-

- ক্যারেক্টার কন্সট্যান্ট (Character Constant)
- ইন্টিজার কন্সট্যান্ট (Integer Constant)
- স্ট্রিং কন্সট্যান্ট (String Constant) ইত্যাদি।

কন্সট্যান্ট ডিক্লারেশন (Constant Declaration) ঃ

কন্সট্যান্ট দুই উপায়ে ডিক্লেয়ার করা যায়। যথা ঃ

- define কীওয়ার্ড ব্যবহার করে। যেমন–

  # define ConstantName Value

  উদাহরণ ঃ # define PI 3.1416

- const কীওয়ার্ড ব্যবহার করে। যেমন–

  const DataType ConstantName = Value;

  উদাহরণ ঃ

  ```
  const float PI = 3.1416;
  ```

উল্লেখ্য যে, #define কীওয়ার্ড ব্যবহার করে কন্সট্যান্ট ঘোষণা করতে চাইলে সেটি হেডার ফাইলের সাথে আর const কীওয়ার্ড ব্যবহার করে করতে চাইলে তা main ( ) ফাংশনের মধ্যে ঘোষণা করতে হবে।

অপারেটর (Operator) ঃ সি ভাষায় বিদ্যমান সকল অপারেটরই সি++ এ ব্যবহৃত হয়। যেমন-

- ম্যাথমেটিক্যাল অপারেটর ঃ (+, –, *, /, %, ++, – – ইত্যাদি)
- রিলেশনাল অপারেটর ঃ (>, > =, <, < =, = =, ! = ইত্যাদি)
- লজিকাল অপারেটর ঃ (&&, !!, ! ইত্যাদি)
- এসাইনমেন্ট অপারেটর ঃ (=)
- কন্ডিশনাল অপারেটর ঃ (? ঃ)
- বিটওয়াইজ অপারেটর ঃ (&, |, ∧, <<, >>, ~ ইত্যাদি।)

উপরোক্ত অপারেটরসমূহের ব্যবহার বিধি ও সি ভাষার ন্যায়।

উদাহরণ ঃ

x = 7, y = 3, z = 5, m = 4, n = 2 হলে,

p = x% y; এই সমীকরণের আউপুট p এর মান হবে 1

p = x++; এই সমীকরণের আউপুট p এর মান হবে 8

p = (x *2) < (y * z); এই সমীকরণের আউপুট p এর মান হবে True অথবা 1

p = (x > p) && (m < n); এই সমীকরণের আউপুট p এর মান হবে False অথবা 0

p = (m >n)? m : n; এই সমীকরণের আউপুট p এর মান হবে 4

C++ স্পেশাল অপারেটর (C ++ Special Operator) ঃ

উপরোক্ত অপারেটরগুলো ছাড়াও C++ ভাষার নিজস্ব অতিরিক্ত বিশেষ কতগুলো অপারেটর আছে। যেমন ঃ <<, >>, : :, :: *, ->*, .*, new, delete, endl, ws ইত্যাদি। নিম্নে C++ এর নিজস্ব অপারেটরগুলোর নাম এবং ব্যবহার উল্লেখ করা হল।

অপারেটর	নাম	কাজ
<<	ইনসারশন বা আউটপুট অপারেটর	ভেরিয়েবলের মান আউটপুট ডিভাইসে পাঠানোর জন্য।
>>	এক্সট্রাকশন বা ইনপুট অপারেটর	কী বোর্ডের মাধ্যমে কোন ভেরিয়েবলের মান ইনপুট করার জন্য।
::	স্কোপ রেজুলুশন অপারেটর	মেম্বার ভেরিয়েবল বা মেম্বার ফাংশন কোন ক্লাসের অন্তর্ভুক্ত তা নির্দেশ করার জন্য।
::*	পয়েন্টার টু মেম্বার অপারেটর	ক্লাসের কোন মেম্বার ভেরিয়েবলের বা মেম্বার ফাংশন নির্দেশ করার জন্য।
->*	পয়েন্টার টু মেম্বার অপারেটর	ক্লাসের মেম্বার ভেরিয়েবল বা মেম্বার ফাংশন নির্দেশ করার জন্য।
.*	পয়েন্টার টু মেম্বার অপারেটর	ক্লাসের কোন মেম্বার ভেরিয়েবল বা মেম্বার ফাংশন নির্দেশ করার জন্য।
new	মেমোরি এলোকেটর অপারেটর	ভেরিয়েবল বা অবজেক্ট ঘোষণার সময় তার জন্য প্রয়োজনীয় মেমোরি স্পেস বরাদ্দ করার জন্য।
delete	মেমোরি ডিলিট অপারেটর	ভেরিয়েবল বা অবজেক্টের জন্য বরাদ্দকৃত মেমোরি স্পেস খালি করার জন্য।
endl	নিউ লাইন অপারেটর	প্রোগ্রামের আউটপুট পরবর্তী লাইনের শুরুতে প্রদর্শন করার জন্য।
setw	ফিল্ড ওয়াইডথ অপারেটর	আউটপুট ভেরিয়েবলের কারেক্টার দৈর্ঘ্য নির্ধারণ করার জন্য।

কন্ট্রোল স্টেটমেন্ট (Control Statements) ঃ

সি++ ভাষার কন্ট্রোল স্টেটমেন্টসমূহ দুই প্রকার। যথা ঃ

- ব্রাঞ্চিং স্টেটমেন্ট ঃ যেমন– if, if-else, nested if-else, else if, switch ইত্যাদি;
- লুপিং স্টেটমেন্ট ঃ যেমন– for, while, do-while ইত্যাদি।

নিম্নে উল্লিখিত স্টেটমেন্টসমূহের গঠন ও উদাহরণ উল্লেখ করা হল।

if স্টেটমেন্টের গঠন ও উদাহরণ ঃ

গঠন ঃ

```
if (condition)
statement;
```

উদাহরণ ঃ

```
if (p>q)
cout << "p is greater than q";
```

if-else স্টেটমেন্টের গঠন ও উদাহরণ ঃ

গঠন ঃ

```
if (condition)
{ statement (s);
}
else
{statement (s);
}
```

উদাহরণ ঃ

```
if (p>q)
{
cout <<"p is greater than q";
}
else
{
cout <<" q is greater than p";
}
```

else-if স্টেটমেন্টের গঠন ও উদাহরণ ঃ

গঠন ঃ

```
if (condition1)
{statement (s);
}
else if (condition2)
{ statement (s);
}
------
------
------
------
else
{statement (s);
}
```

উদাহরণ ঃ

```
if (Total Marks > = 600)
{ cout <<" Result is First Division";
}
else if (Total Marks > = 450)
{ cout << "Result is Second Division";
}
else if (Total Marks> = 330)
{ cout <<"Result is Third Division";
}
else
{
cout <<" Result is Fail";
}
```

Nested if- else স্টেটমেন্টের গঠন ও উদাহরণ ঃ

গঠন ঃ

```
if (condition1)
{ if (condition2)
statement (s);
else
statement (s);
}
else
statement (s);
```

উদাহরণ ঃ

```
if (a > b)
{if (a>c)
cout <<" Largest number is a";
else
cout << "Largest number is c";
}
else
{if (c>b)
cout <<"Largest number is c";
else
cout <<" Largest number is b" ;
}
```

switch স্টেটমেন্টের গঠন ও উদাহরণ ঃ

গঠন ঃ

```
DataType IndexVariable;
switch (expression)
{case1:
{statement1;
break;
}
case2 :
{statement2;
break;
}
case3 :
{
{statement3;
break;
}
default :
{statement n;
}
}
```

উদাহরণ ঃ

```
char ch;
ch = getch ( );
switch (ch)
{ case1:
{
add ( );
break;
}
case2 :
{
sub ( );
break;
}
case3: ;
{
mul ( );
break;
}
case4 :
{
div ( );
break;
}
}
```

for লুপের গঠন ও উদাহরণ ঃ

গঠন ঃ

```
for (Initialization; Condition; Increment/Decrement)
{Statement (s);
}
```

উদাহরণ ঃ

```
for (i = 5; i < = 100; i = i + 5)
sum = sum + i;
```

while স্টেটমেন্টের গঠন ও উদাহরণ ঃ

গঠন ঃ

```
Initialization;
while (Condition)
{Statement (s);
Increment/Decrement;
}
```

উদাহরণ ঃ

```
i = 1;
while (i < = 5)
{
cout << i;
i++;
}
```

do-while স্টেটমেন্টের গঠন ও উদাহরণ ঃ

গঠন ঃ

```
Initialization;
do
{Statement (s);
Increment/Decrement;
} While (Condition);
```

উদাহরণ ঃ

```
i = 1;
do
{
cout <<i;
i++;
} while (i < = 5);
```

৪.৪.১ সি++ ফাংশন (Function in C++) ঃ

সি++ ভাষায় ফাংশন একটি অতি গুরুত্বপূর্ণ বিষয়। ফাংশন ভাল না বুঝলে সি++ ভাষার কোন কিছুই বুঝা যাবে না। কারণ ক্লাস, কনস্ট্রাক্টর, কপি কনস্ট্রাক্টর, ডেস্ট্রাক্টর, ফাংশন ওভারলোডিং, ফ্রেন্ড ফাংশন, ইনহেরিটেন্স ইত্যাদি সবকিছুতেই ফাংশন নিয়ে কাজ করতে হয়। তাই প্রোগ্রামে কিভাবে ফাংশন ব্যবহার করা হয় তা আগে জেনে নিতে হবে।

যেকোন ইউজার ডিফাইন্ড ফাংশনে তিনটি অংশ বিদ্যমান। যথা ঃ

- ফাংশন ডিক্লারেশন (Function declaration)
- ফাংশন ডেফিনিশন (Function definition) ও
- ফাংশন কলিং (Function calling)।

'ফাংশন ডিক্লারেশন' এর ফরম্যাট হল ঃ

```
ReturnType FunctionName ( );
```

উদাহরণ ঃ

```
void add ( );
void TriangleArea ( ); ইত্যাদি।
```

'ফাংশন ডেফিনিশন' এর ফরম্যাট নিম্নরূপ ঃ

```
ReturnType FunctionName ( )
{
//Function body
}
```

উদাহরণ ঃ

```
void add ( )
{ int a, b result;
cout <<"Enter the value of a, b :";
cin >>a >>b;
result = a + b;
cout <<"Result is = "<< result;
}
```

'ফাংশন কলিং' এর ফরম্যাট হল ঃ

```
FunctionName ( );
```

উদাহরণ ঃ

```
add ( );
```

মনে রাখতে হবে যে, একই প্রোগ্রামে এক বা একাধিক ইউজার ডিফাইন্ড ফাংশন ব্যবহৃত হতে পারে। সেক্ষেত্রে

- হেডার ফাইল ঘোষণার পর পরই সাধারণত "ফাংশন বা ফাংশনসমূহের ডিক্লারেশন" হবে
- প্রতিটি ইউজার ডিফাইন্ড ফাংশনকে আলাদা আলাদা ডিফাইন করতে হবে। তবে ফাংশন ডেফিনিশন অংশটুকু main ( ) ফাংশনের আগে অথবা পরে হবে।
- 'ফাংশন কলিং' এর ক্ষেত্রে প্রয়োজনীয় ফাংশন বা ফাংশনসমূহকে main ( ) ফাংশনের অভ্যন্তরে কল (call) করতে হবে।

নিম্নে ফাংশন ব্যবহার করে দুটি সংখ্যা যোগ করার একটি প্রোগ্রাম লিখা হল ঃ

```
# include <iostream.h>
# include <conio.h>
void add ( );
void add ( )
{ int a, b, result;
cout <<"Enter the value of a, b :";
cin>>a>>b;
result = a + b;
cout <"summation is = "<<result;
}
void main ( )
{clrscr ( );
add ( );
getch ( );
}
```

৪.৫ সি++ ল্যাংগুয়েজে ক্লাস (Features of classes in C++) ঃ

সি$^{++}$এর একটি অতি গুরুত্বপূর্ণ বৈশিষ্ট্য হল ক্লাস (Class)। ক্লাস হল স্ট্রাকচারের বর্ধিত রূপ এবং স্ট্রাকচারের মতই এক ধরনের ইউজার ডিফাইন্ড ডাটা টাইপ। কিন্তু স্ট্রাকচারের বেশকিছু সীমাবদ্ধতা রয়েছে। যেমন-

স্ট্রাকচার ভেরিয়েবলকে পুরোপুরিভাবে বিল্ট-ইন ডাটা টাইপ ভেরিয়েবলের মত ব্যবহার করা যায় না। সব সময় দুটো স্ট্রাকচার ভেরিয়েবলকে সরাসরি যোগ, বিয়োগ, গুণ, ভাগ ইত্যাদি করা যায় না। আবার স্ট্রাকচারের মেম্বার ভেরিয়েবলগুলো সাধারণত গ্লোবাল প্রকৃতির। স্ট্রাকচার ভেরিয়েবলের ডাটাকে বাইরের অন্য কোন প্রোগ্রাম বা ফাংশনের হস্তক্ষেপ থেকে লুকানো যায় না। ফলে স্ট্রাকচার ভেরিয়েবলের যে কোন ডাটা মেম্বারকে প্রোগ্রামের অন্য কোন ফাংশন দ্বারা পরিবর্তন বা পরিবর্ধন করা যায়। এতে সিস্টেমে ব্যবহৃত ডাটার গোপনীয়তা ও বিশ্বস্ততা হ্রাস পায়। কিন্তু ক্লাসের কোন ডাটা মেম্বারকে প্রাইভেট ঘোষণার মাধ্যমে বাইরের অন্য কোন প্রোগ্রাম বা ফাংশনের হস্তক্ষেপ থেকে রক্ষা করা যায়। ফলে সিস্টেমে ব্যবহৃত ডাটার গোপনীয়তা ও বিশ্বস্ততা বৃদ্ধি পায়। এসব বিবেচনা করে সি++ এ স্ট্রাকচারের বর্ধিতরূপে ক্লাস ব্যবহার করা হয়।

ক্লাস (class) একটি ইউজার ডিফাইন্ড ডাটা টাইপ। ক্লাসে কিছু ভেরিয়েবল এবং ঐ ভেরিয়েবলগুলো এক্সেস (access) করার জন্য কিছু ফাংশন থাকে। ক্লাসে ব্যবহৃত ভেরিয়েবলকে মেম্বার ভেরিয়েবল (member variable) বা ডাটা মেম্বার (Data member) এবং মেম্বার ভেরিয়েবলের সাথে সম্পর্কিত ফাংশনকে মেম্বার ফাংশন (Member function) বা মেথড (method) বলা হয়। মেম্বার ভেরিয়েবল এবং মেম্বার ফাংশনসমূহকে একযোগে ক্লাস মেম্বার (class member) বলে। ক্লাস মেম্বার হিসাবে বৈধ ডাটা টাইপের যে কোন সংখ্যক ভেরিয়েবল বা অন্য কোন ক্লাস ব্যবহৃত হতে পারে।

সি++ প্রোগ্রামিং ভাষায় প্রোগ্রাম রচনার নিম্নলিখিত ধাপসমূহ অনুসরণ করতে হয়। যথা ঃ

- ক্লাস ডিক্লারেশন (Class declaration)
- মেম্বার ফাংশন ডেফিনিশন (Member function definition)
- ক্লাস অবজেক্ট ডিক্লারেশন (Class object declaration) এবং
- মেম্বার ফাংশন কলিং (Member function calling)।

ক্লাস ডিক্লারেশন (Class declaration) ঃ ক্লাস ঘোষণার জন্য class কীওয়ার্ডের পরে ঐ ক্লাসের নাম দিতে হয়, ক্লাস বডি দ্বিতীয় বন্ধনী '{ }' দ্বারা সীমাবদ্ধ থাকে এবং সেমিকোলন (;) দ্বারা শেষ হয়। ক্লাস বডিতে ক্লাসের মেম্বার ভেরিয়েবল এবং মেম্বার ফাংশনগুলো তাদের এক্সেসিবিলিটি মোডসহ উল্লেখ করতে হয়। ক্লাসের মেম্বার ভেরিয়েবল এবং মেম্বার ফাংশনের জন্য তিন ধরনের এক্সেসিবিলিটি মোড হতে পারে ঃ যথা প্রাইভেট (private), পাবলিক (public) এবং প্রটেকটেড (protected)। এদেরকে ক্লাসের মেম্বারসমূহের ভিসিবিলিটি মোডও বলা হয়। ক্লাসের প্রাইভেট, পাবলিক এবং প্রটেকটেড মেম্বারসমূহ যে কোন ক্রমে যে কোন সংখ্যক বার থাকতে পারে। ক্লাসের মেম্বারের জন্য কোন মোড ঘোষণা করার পর অন্য কোন মোড উল্লেখ না করা পর্যন্ত ঐ মোড বিবেচিত হয়। ক্লাসে মেম্বারের জন্য কোন মোড উল্লেখ না থাকলে তা প্রাইভেট হিসেবে গণ্য হয়। ক্লাসে কোন পাবলিক ডাটা মেম্বার না থাকলে প্রাইভেট ডাটা মেম্বারগুলোকে এক্সেস করার জন্য ন্যূনতম একটি পাবলিক মেম্বার ফাংশন থাকা আবশ্যক। ক্লাস ঘোষণার সাধারণ ফরম্যাট হল ঃ

```
class UserDefinedName
{                          ←──────  ক্লাসের নাম
private :
Variable Decleration;   ┐
Function Decleration;   ┘ ← প্রাইভেট ঘোষণা
protected:
Variable Decleration;   ┐
Function Decleration;   ┘ ← প্রটেকটেড ঘোষণা
public :
Variable Decleration;   ┐
Function Decleration;   ┘ ← পাবলিক ঘোষণা
};        // end of class (; অবশ্যই ব্যবহার করতে হবে।)
```

উদাহরণ ঃ

```
class Student  {
        // By default private accessibility mode
    long        Roll;
    char        Name [40];
protected :     // protected accessibility mode
    float       Marks;
public :        // public accessibility mode
     void GetData ( );
     void Display ( );
        };
```

উপরোক্ত ক্লাসের নাম দেয়া হয়েছে Student। ক্লাসের নাম একটি আউডেন্টিফায়ার, সুতরাং আইডেন্টিফায়ার নামকরণের নিয়মানুযায়ী ক্লাসের যে কোন নাম দেওয়া যেতে পারে। এখানে Student নামে একটি নতুন ডাটা টাইপ তৈরি হল, যা এই টাইপের কোন অবজেক্ট ঘোষণার জন্য ব্যবহৃত হবে। এখানে Name ও Roll ডাটা মেম্বার ডিফল্ট হিসেবে প্রাইভেট, Marks ডাটা মেম্বার প্রটেকটেড এবং GetData ( ) ও Display ( ) মেম্বার ফাংশন পাবলিক মোডে ঘোষিত হয়েছে।

মেম্বার ফাংশন ডেফিনিশন (Member function definition) ঃ ফাংশন ডেফিনিশন বলতে সংশ্লিষ্ট ফাংশনটি যে সুনির্দিষ্ট অপারেশন সম্পন্ন করার জন্য ডিক্লেয়ার করা হয়েছে তার জন্য প্রয়োজনীয় কোড লিখাকে বুঝায়। একে ফাংশন বডিও বলে। সি++ ভাষায় ক্লাসের মেম্বার ফাংশনসমূহকে দুই উপায়ে ডিফাইন করা যায়। যথা ঃ

- ক্লাসের ভিতরে (Inside the class)
- ক্লাসের বাইরে (Out side the class)।

নিম্নে ক্লাসের ভিতরে মেম্বার ফাংশন ডিফাইন করার ফরম্যাট দেখানো হল ঃ

```
class Student {
   long        Roll;
   char        Name [40];
protected :
   float       Marks;
public :
void GetData ( ) // Inside Member Function Definition
{
     // Function Body
}
void Display ( ) // Inside Member Function Definition
{
    // Function Body
}
}; // End of class Decleration
```

অপরদিকে ক্লাসের বাইরে মেম্বার ফাংশন ডিফাইন করার ফরম্যাট হল নিম্নরূপ।

```
ReturnType ClassName : : FunctionName ( )
{
// Function Body
}
```

উদাহরণ ঃ

```
class Student {
private :
            long   Roll;
            char   Name [40];
protected :
            float   Marks;
public :
            void GetData ( )
            void Display ( ); // Function Definition
}; // End if class
void Student : : GetData ( )
   {
   // Function Body
   }
void Student : : Display ( )
   {
   // Function Body
   }
```

ক্লাস অবজেক্ট ডিক্লারেশন (Class object declaration) ঃ ক্লাসে ডিক্লেয়ারকৃত মেম্বার ভেরিয়েবল ও মেম্বার ফাংশনসমূহকে এক্সেস (Access) করার জন্য সংশ্লিষ্ট ক্লাসের অবজেক্ট ঘোষণা করতে হয়। ক্লাস অবজেক্টকে দুই ভাবে ডিক্লেয়ার করা যায়। যথা ঃ

- ক্লাস ঘোষণার সময় এবং
- main ( ) ফাংশনের অভ্যন্তরে।

ক্লাস ঘোষণার সময় অবজেক্ট ডিক্লেয়ার করার ফরম্যাট নিম্নরূপ ঃ

```
class ClassName {
public :
private : // class body
}ObjectName;
```

উদাহরণ ঃ

```
class Student  {
private :
        long Roll;
        char Name [40];
protected :
        float Marks;
public :
        void GetData ( )
        {
        // Function Body
        }
        void Display ( )
        {
        // Function Body
        }
}S1;    // S1 is an object of Student class
```

main ( ) ফাংশনের অভ্যন্তরে অবজেক্ট ডিক্লারেশনের ফরম্যাট হল ঃ

```
ClassName ObjectName;
```

উদাহরণ ঃ

```
void main ( )
{clrscr ( );
Student S; // Object Declaration
------
------
------
------
getch ( );
}
```

মেম্বার ফাংশন কলিং (Member function calling) ঃ ক্লাসে ব্যবহৃত মেম্বার ফাংশনসমূহ main ( ) ফাংশনের ভিতরে কল (Call) করতে হয়। এজন্য প্রথমে সংশ্লিষ্ট ক্লাসের জন্য অবজেক্ট ডিক্লেয়ার করতে হয়। পরবর্তীতে ঐ অবজেক্ট সহযোগে মেম্বার ফাংশন বা ফাংশনসমূহকে কল করতে হয়। মেম্বার ফাংশন কল (call) করার ফরম্যাট নিম্নরূপ ঃ

```
ObjectName. FunctionName ( );
```

উদাহরণ ঃ

```
void main ( )
{clrscr ( );
student s;
s.Get data ( );// মেম্বার ফাংশন কলিং
s. Display ( );// মেম্বার ফাংশন কলিং
getch ( );
}
```

নিম্নে উপরোক্ত ৪টি ধাপ অনুসরণ করে সি++ ভাষায় ক্লাস ব্যবহার করে আয়তক্ষেত্রের ক্ষেত্রফল নির্ণয়ের একটি প্রোগ্রাম লেখা হল ঃ

```
#include<iostream.h>
#include<conio.h>
class Area{
public:
int length,width,Area;
void RectangleArea();
};
void Area::RectangleArea()
{
cout<<"Enter the value of length,width:";
cin>>length>>width;
Area=length*width;
cout<<"Rectangle Area is ="<<Area;
}
void main()
{
clrscr();
Area A;
A.RectangleArea();
getch();
}
```

তবে private ও protected অংশের মেম্বার ফাংশনকে সরাসরি অবজেক্ট ব্যবহার করে কল করা যায় না। সেক্ষেত্রে উল্লিখিত দুই অংশের ফাংশনসমূহকে প্রথমে কোন public ফাংশনের ডেফিনিশন অংশে কল করতে হয়।

৪.৬ সি++ ভাষায় ফাংশন ওভারলোডিং (Function overloading in C++) ঃ

যে প্রক্রিয়ায় সি++ ভাষায় একই নামের দুই বা ততোধিক ফাংশন ব্যবহার করে কোন সমস্যা ছাড়াই ভিন্ন ভিন্ন কা সম্পন্ন করা যায় তাকে ফাংশন ওভারলোডিং (Function overloading) বলে।

এ প্রক্রিয়ায় একই প্রোগ্রামের একাধিক ফাংশনের নাম একই হবে কিন্তু তাদের প্যারামিটার (Parameter), প্যারামিটার সংখ্যা, প্যারামিটার ডাটা টাইপ অবশ্যই ভিন্ন ভিন্ন হবে। ফাংশন কল (Call) করে যখন আউটপুট প্রসেস করবে তখন ঐসব ফাংশনের argument সংখ্যা বা ডাটা টাইপের উপর ভিত্তি করে কম্পাইলার প্রোগ্রামটি execute করবে।

উদাহরণ ঃ

```
void show ( );
void show (int i);
void show (int i, char ch);
```

নিম্নে ফাংশন ওভারলোডিং প্রক্রিয়া একটি প্রোগ্রামের মাধ্যমে দেখানো হল ঃ

```
#include<iostream.h>
#include<conio.h>
#include<math.h>
class Result
{
public:
void add(int x,int y);
void add(int a);
};
void Result::add(int x, int y)
{
cout<<"x+y is ="<<x+y;
}
void Result::add(int a)
{
cout<<endl<<"Result="<<a+a;
}
void main()
{
clrscr();
Result R;
int m,n;
cout<<"Enter the value of m,n:";
cin>>m>>n;
R.add(m,n);
R.add(m);
getch();
}
```

ফাংশন ওভারলোডিং প্রক্রিয়ার নিয়মাবলি বা শর্তসমূহ ঃ

নিম্নবর্ণিত নিয়মাবলি বা শর্তসমূহ প্রতিপালন সাপেক্ষে সি++ ভাষায় ফাংশন ওভারলোডিং প্রক্রিয়া সম্পন্ন হয়। যথা ঃ

- একাধিক ফাংশনের নাম একই (same) হলেও এর প্যারামিটার বা প্যারামিটার সংখ্যা অবশ্যই ভিন্ন হবে।
- ফাংশন ওভারলোডিং প্রক্রিয়ায় ব্যবহৃত ফাংশনসমূহ একই ক্লাসের মেম্বার ফাংশন হতে হবে।
- একই নামের একাধিক ফাংশনের রিটার্ন টাইপ ভিন্ন ভিন্ন হলেও যদি ফাংশনসমূহের প্যারামিটার সংখ্যা এবং প্যারামিটার টাইপ একই (same) হয় তবে তারা overloaded function হিসেবে কাজ করবে না। যেমন–

```
int sum (int a, int b);   ||
float sum (int a, int b); ||  Overloaded function নয়
```

- একই নামের একাধিক ফাংশনের রিটার্ন টাইপ একই (same) হলেও যদি ফাংশনসমূহের প্যারামিটার সংখ্যা এবং প্যারামিটার টাইপ ভিন্ন হয় তবে তারা overloaded function হিসেবে কাজ করবে। যেমন-

```
int sum (int a, int b);       ||
int sum (float a, float b);   ||  overloaded function.
```

- কোন ফাংশন ওভারলোডেড হবে কিনা তা শুধুমাত্র প্যারামিটারের উপর নির্ভর করে। রিটার্ন ডাটা টাইপের সাথে এর কোন সম্পর্ক নেই।
- একই টাইপ এর argument এবং সমসংখ্যক reference argument বিশিষ্ট একাধিক ফাংশন নিয়ে overloading করা যাবে না। এমনকি তাদের return type ভিন্ন হলেও। যেমন-

```
int sum (int arg);      ||
float sum (int & arg)   ||  overloaded function নয়
```

- ভিন্ন ভিন্ন সাইজের অ্যারে বিশিষ্ট সমসংখ্যক আরগুমেন্ট বিশিষ্ট একাধিক ফাংশন ও Overloaded function নয়। এমনকি তাদের return type ভিন্ন হলেও। যেমন-

```
int sum (int arg [20];   ||
float sum (int arg [40]; ||  overloaded function নয়।
```

ফাংশন ওভারলোডিং এর সীমাবদ্ধতা ঃ ফাংশন ওভারলোডিং প্রক্রিয়ার ক্ষেত্রে বেশকিছু সীমাবদ্ধতা রয়েছে। যেমন-

- ফাংশন ওভারলোডিং প্রক্রিয়ার প্রধান সীমাবদ্ধতা হল এতে ব্যবহৃত ফাংশনগুলোর ক্ষেত্রে returntype এর কোন ভূমিকা নেই।
- একই গ্রুপের একাধিক ফাংশন একটি কমন ঘোষণার মাধ্যমে ব্যবহার করা যায় না।
- একই নামে ভিন্ন ভিন্ন arguments এর একাধিক ফাংশন ব্যবহার করা গেলেও প্রতিটি আলাদাভাবে ডিক্লেয়ার ও ডিফাইন করতে হয়।
- এখনো সব ধরনের কম্পাইলার ফাংশন ওভারলোডিং প্রক্রিয়া অনুমোদন করে না।

৪.৬.১ অপারেটর ওভারলোডিং (Operator overloading) ঃ

অপারেটর ওভারলোডিং সি++ ভাষার একটি গুরুত্বপূর্ণ বৈশিষ্ট্য। কোন অপারেটর এর বিল্ট ইন (Built in) যে কাজ করার ক্ষমতা থাকে তার চেয়ে বাড়তি কিছু কাজ করার ক্ষমতা তৈরির প্রক্রিয়াকে অপারেটর ওভারলোডিং বলে। যেমন- প্লাস অপারেটর (+) সাধারণত দুটি সংখ্যা যোগ করার কাজে ব্যবহৃত। কিন্তু অপারেটর ওভারলোডিং প্রক্রিয়ায় প্লাস অপারেটর (+) দুটি সংখ্যা যোগ করার পাশাপাশি দুটি স্ট্রিং (String) এর যোগফল হিসেবে অপর একটি স্ট্রিং আউটপুট প্রদান করতে পারে।

অপারেটর ওভারলোডিং, প্রক্রিয়ায় নতুন কোন অপারেটর তৈরি করা সম্ভব নয়। কেবল বিদ্যমান অপারেটরসমূহের পরিধি বৃদ্ধি করা যায়। আবার অপারেটর ওভারলোডিং এর মাধ্যমে বিদ্যমান অপারেটরসমূহের প্রচলিত অর্থ পরিবর্তন করা যায় না। যেমন- দুটি সংখ্যা বা স্ট্রিং অবজেক্ট যোগ করার জন্য মাইনাস অপারেটর (–) ওভারলোডিং করা যায় না।

সি++ প্রোগ্রামিং পদ্ধতিতে সি++ এর প্রায় সব অপারেটরকেই ওভারলোডিং করা সম্ভব। যেমন–

+	-	*	/	%	^	!	= <	>	+ =	- =	^=	&=	|=
<<	>>	<<=	<=	>=	&&	||	++	- -	( )	[ ]	&	|~	*=
/=	%=	>>=	= =	!=	,	->	-=>*	new	এবং delete				

তবে বিশেষ কিছু অপারেটর আছে যেগুলোর ওভারলোডিং প্রক্রিয়ায় ব্যবহার করা সম্ভব নয়। যেমন- ডাইরেক্ট মেম্বার অপারেটর (.), ডাইরেক্ট পয়েন্টার অপারেটর (·*), কন্ডিশনাল অপারেটর (: ?), স্কোপ রেজুলুশন অপারেটর (: :), বাইট সাইজ অপারেটর (size of) এবং প্রিপ্রসেসর অপারেটর (#, # #) ইত্যাদি।

অপারেটর ওভারলোডিং এর নিয়ম ঃ

- কেবল বিদ্যমান অপারেটরের জন্যই ওভার লোডিং প্রযোজ্য। নতুন কোন অপারেটর তৈরি করা যায় না।।
- বিদ্যমান অপারেটরসমূহের প্রচলিত অর্থ পরিবর্তন করা যায় না। যেমন ঃ দুইটি সংখ্যা যোগ করার জন্য মাইনাস (–) অপারেটর ওভার লোডিং করা যায় না।
- Overloaded operator শুধুমাত্র সংশ্লষ্ট Class object এর ক্ষেত্রেই কাজ করে।
- ওভারলোডেড অপারেটরের মধ্যে Custom data type এর নূন্যতম একটি Operand থাকতে হবে।
- কতগুলো Operator overload করার জন্য ফ্রেন্ড ফাংশন ব্যবহার করা যায় না। যেমন– Assignment operator (=), function call (()), class member Access (.) operator ইত্যাদি।
- Binary operator এর Value return করার ক্ষেত্রে এক্ষেত্রে argument পরিবর্তন করা যায় না।
- unary operator সমূহের জন্য প্রত্যক্ষ কোন argument প্রয়োজন হয় না।

অপারেটর ওভারলোডিং এর ধাপসমূহ ঃ অপারেটর ওভারলোডিং এর জন্য নিম্নবর্ণিত ধাপসমূহ অনুসরণ করতে হয়।

- প্রথমে ওভারলোডিং অপারেশনের জন্য প্রয়োজনীয় ডাটা সমন্বয়ে ক্লাস (class) ডিক্লেয়ার করতে হবে।
- এবার ক্লাসের পাবলিক সেকশনে অপারেটর ওভারলোডেড ফাংশন (operator overloaded function) ডিক্লেয়ার করতে হবে।
- সবশেষে ঘোষিত অপারেটর ওভারলোডেড ফাংশনটি ডিফাইন করতে হবে। অর্থাৎ ফাংশন বডি লিখতে হবে।

অপারেটর ওভারলোডিং ফাংশন ডিক্লারেশন ঃ অপারেটর ওভারলোডিং ফাংশন ডিক্লারেশনের ফরম্যাট হল ঃ

```
ReturnType ClassName : : operatorOperatorToBeOverloaded (ArgList)
{
// Function body that defines the task
}
```

এখানে, ReturnType হল সি++ এর যে কোন বৈধ রিটার্ন টাইপ, এরপর operator কীওয়ার্ডটি থাকবে এবং তার নাম OperatorToBeOverloaded যে অপারেটরটি ওভারলোড করতে হবে আর ArgList ওভারলোডেড ফাংশনে বর্ণিত আরগুমেন্ট তালিকা।

উদাহরণ ঃ

```
int sum : : operator + (sum s)
{
sum T;
T. value = value + s. value;
return (T);
}
```

অপারেটর ওভারলোডিং ফাংশন কলিং ঃ ইউনারি অপারেটরের জন্য মেম্বার ফাংশনের ওভালোডেড অপারেটর ফাংশন কল করার ফরম্যাট হচ্ছে ঃ

```
OperatorToBeOverloaded  Argument;
```

বা,

```
Argument  OperatorToBeOverloaded;
```

যা ফ্রেন্ড ফাংশনের বেলায় operator OperatorToBeOverloaded (Argument); হিসেবে গণ্য করা হয়।

আবার, বাইনারি অপারেটরের জন্য মেম্বর ফাংশনের অভারলোডেড অপারেটর ফাংশন কল করার ফরম্যাট হচ্ছে ঃ

```
Argument1 OperatorToBeOverloaded Argument2;
```

যা ফ্রেন্ড ফাংশনের ক্ষেত্রে

```
operator OperatorToBeOverloaded (Argument1, Argument2);
```

হিসেবে গণ্য করা হয়। উল্লেখ্য, যদি কখনো কোন ওভারলোডেড ফাংশনে ফ্রেন্ড ফাংশন এবং মেম্বার ফাংশন উভয়ই থাকে তবে আরগুমেন্ট-এ ধরন অনুযায়ী কম্পাইলার উপযুক্ত ফাংশন কল করে।

নিম্নে অপারেটর ওভারলোডিং ব্যবহার করে একটি প্রোগ্রাম দেয়া হল ঃ

```
#include<iostream.h>
#include<conio.h>
class Op_Over
{
private:
int x,y;
public:
void Operator();
void GetData(int a,int b);
void Show();
};
```

```
void Op_Over::GetData(int a,int b)
{
x=a;y=b;
}
void Op_Over::Show()
{
cout<<x<<" ";
cout<<y<<" ";
}
void Op_Over::Operator()
{
x=-x;y=-y;
cout<<x<<" "<<y;
}
void main()
{
clrscr();
Op_Over A;
A.GetData(7,-9);
cout<<"A:";
A.Show();
cout<<endl<<"Operator Overloading of A:";
A.Operator();
getch();
}
```

৪.৭ সি++ ভাষায় কনস্ট্রাক্টর, কপি কনস্ট্রাক্টর, ডেস্ট্রাক্টর ও ফ্রেন্ড ফাংশন (Constructor, copy constractor, destructor and friend function in C++) ঃ

কনস্ট্রাক্টর (Constructor) ঃ কনস্ট্রাক্টর হল এমন এক বিশেষ ধরনের ফাংশন যা কোন অবজেক্ট ঘোষণার সময় তার মান নির্ধারণ করে এবং মেমোরি স্পেস বরাদ্দ করে। ক্লাসে ব্যবহৃত মেম্বার ভেরিয়েবলগুলোর মান নির্ধারণ ও মেমোরি স্পেস বরাদ্দ করে বিধায় একে কনস্ট্রাক্টর (Constructor) বলে।

উল্লেখ্য যে,

- কনস্ট্রাক্টর ফাংশনের নাম ও ক্লাসের নাম একই (same) হবে।
- কনস্ট্রাক্টর ফাংশনকে অবশ্যই ক্লাসের পাবলিক (Public) অংশে ডিক্লেয়ার করতে হবে।
- কোন Return Type ব্যবহার করা যায় না এমনকি void ও নয়।

কনস্ট্রাক্টর ফাংশনকেও দুই ভাবে ডিফাইন করা যায়। যথা ঃ

- ক্লাসের ভিতরে (Inside the class) এবং
- ক্লাসের বাইরে (Outside the class)।

কোন ক্লাসে একবার কনস্ট্রাক্টর ফাংশন ডিক্লেয়ার করা হলে অবশ্যই তাকে ডিফাইন করতে হবে যদিও তা শূন্য বা কোন মান রিটার্ন করবে না।

কনস্ট্রাক্টর ফাংশন ডিক্লারেশনের ফরম্যাট হল ঃ

```
class ClassName{
private :
        // private members
protected :
        // protected members
public :
        ClassName ( ); // Constructor declaration
//other members
};
ClassName :: ClassName ( )
{
// Constructor function definition
}
```

উদাহরণ ঃ

```
class Student {
private :
        int Roll;
        char Name;
protected :
        float gpa ( );
public :
        void display ( );
        Student ( );// constructor (ফাংশনটির নাম ও ক্লাসের নাম একই)
};
```

এবার Student ( ) নামক কনস্ট্রাক্টর ফাংশনটি ডিফাইন করা হল।

```
Student : : Student ( )
{
//function body
}
```

উপরোক্ত ক্ষেত্রে প্রথম 'Student' হচ্ছে ClassName এবং শেষ 'Student' হচ্ছে ConstructorName। নিম্নে কনস্ট্রাক্টর ফাংশন ব্যবহার করে একটি প্রোগ্রাম লেখা হল।

```
#include<iostream.h>
#include<conio.h>
class Area{
private:
int arm,SquareArea;
public:
Area();
void Show();
};
```

```
Area::Area()
{
cout<<"Enter the value of arm:";
cin>>arm;
SquareArea=arm*arm;
cout<<"Square Area is ="<<SquareArea;
}
void Area::Show()
{
void Area();
}
void main()
{
clrscr();
Area A;
A.Show();
getch();
}
```

কনস্ট্রাক্টর ফাংশনের বৈশিষ্ট্য ঃ

- কনস্ট্রাকটর ফাংশনের নাম তার ধারণকারী ক্লাসের নামের অনুরূপ হয়। অর্থাৎ Class Name ও FunctionName একই (same)।
- কনস্ট্রাকটর ফাংশন ক্লাসের পাবলিক অংশে ঘোষণা করা হয়।
- কনস্ট্রাকটর ফাংশন কোন মান রিটার্ন করতে পারে না। কারণ এর কোন রিটার্ন টাইপ নেই, এমন কি void টাইপও নয়।
- কনস্ট্রাকটর ফাংশন কল করতে হয় না। কোন ক্লাসের অবজেক্ট ঘোষণার সময় কম্পাইলার স্বয়ংক্রিয়ভাবে সেই ক্লাসের কনস্ট্রাকটর ফাংশন আহ্বান করে।
- কনস্ট্রাকটর ফাংশন ইনহেরিট করা যায় না। কোন ক্লাসের অবজেক্ট ঘোষণার সময় কম্পাইলার স্বয়ংক্রিয়ভাবে সেই ক্লাসের কনস্ট্রাকটর ফাংশন আহ্বান করে।
- কনস্ট্রাকটর ফাংশন ইনহেরিট করা যায় না। যদিও কোন ডিরাইভড ক্লাস কনস্ট্রাকটর বেস ক্লাস কনস্ট্রাকটর আহ্বান করতে পারে।
- কনস্ট্রাকটর ফাংশন ডিফল্ট আরগুমেন্ট বিশিষ্ট হতে পারে।
- কনস্ট্রাকটর ফাংশন ভরচুয়াল ফাংশন হতে পারে না।
- কনস্ট্রাকটর ফাংশন বিশিষ্ট অবজেক্টকে ইউনিয়ন (union) এর মেম্বার হিসেবে ব্যবহার করা যায় না।
- কনস্ট্রাকটর ফাংশনকে ইন-লাইন (inline) ফাংশন আকারে বর্ণনা করা যায়।
- কনস্ট্রাকটর ফাংশনের আরগুমেন্ট ডাটা টাইপ সংশ্লিষ্ট ক্লাস বাদে অন্য যে কোন ডাটা টাইপের হতে পারে।

কনস্ট্রাক্টর ফাংশনের প্রয়োজনীয়তা ঃ

অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর একটি অন্যতম উদ্দেশ্য হচ্ছে-এমন একটি user defined ডাটা টাইপ তৈরি করা যা বিল্ট ইন ডাটা টাইপের মত কাজ করবে। বহু সংখ্যক অবজেক্ট বিশিষ্ট কোন ক্লাসের জন্য তার প্রাইভেট মেম্বার ভেরিয়েবলগুলোর মান নির্ধারণের জন্য বার বার সংশ্লিষ্ট মেম্বার ফাংশন কল করা সমস্যাজনক। OOP তে সরাসরি কোন মেম্বার ফাংশন কল করে কোন অবজেক্টের private মেম্বার ভেরিয়েবলের মান নির্ধারণ সম্ভব নয়। এই কাজটি সহজভাবে এবং স্বয়ংক্রিয়ভাবে সম্পন্ন করার জন্য Constructor function ব্যবহার করা হয়। কারণ Constructor function এর সাহায্যে কোন Object ঘোষণার সময় স্বয়ংক্রিয়ভাবে Object গুলোর মান নির্ধারণ করা হয়। ফলে বহু সংখ্যাক Object বিশিষ্ট কোন ক্লাসের private ভেরিয়েবলগুলোর মান নির্ধারণ করার জন্য বার বার সংশ্লিষ্ট মেম্বার ফাংশন কল করার প্রয়োজন হয় না।

কপি কনস্ট্রাক্টর (Copy constructor) ঃ কোন কনস্ট্রাক্টর ফাংশনের আরগুমেন্ট হিসাবে একই ক্লাসের কোন রেফারেন্স অ্যাড্রেস ব্যবহৃত হলে তাকে কপি কনস্ট্রাক্টর (Copy constructor) বলে। কপি কনস্ট্রাক্টরের ক্ষেত্রে একই নামে একাধিক কনস্ট্রাক্টর ফাংশন ব্যবহৃত হয় বলে একে ওভারলোডিং কনস্ট্রাক্টরও বলে।

একটি নির্দিষ্ট অবজেক্ট থেকে অন্য এক বা একাধিক অবজেক্টের মান নির্ধারণের জন্য কপি কনস্ট্রাক্টর ব্যবহৃত হয়। যেমন–

```
Student S2 = S1;
```

অথবা

```
Student S2 (S1);
```

স্টেটমেন্টটি Student টাইপের একটি অবজেক্ট S2 বর্ণনাপূর্বক S1 অবজেক্টের মানকে S2 অবজেক্টের মান হিসেবে নেয় যেখানে S1 একই টাইপের পূর্বে ঘোষিত কোন অবজেক্ট। কিন্তু S1 এবং S2 অবজেক্ট দুটোই Student টাইপের হলেও S2 = S1; কোন বৈধ কপি কনস্ট্রাক্টর স্টেটমেন্ট নয়। কারণ এটি কেবল ওভারলোডেড অ্যাসাইনমেন্ট-এর ক্রিয়া সম্পাদন করে অর্থাৎ S1 অবজেক্টের মান S2 অবজেক্টে পাঠায় কিন্তু কোন অবজেক্ট তৈরি করতে পারে না। কপি কনস্ট্রাকটরের মাধ্যমে এভাবে কোন অবজেক্টের মান নির্ধারণ করার প্রক্রিয়াকে কপি ইনিশিয়ালাইজেশন (Copy Initialization) বলা হয়। উল্লেখ্য, একটি কপি কনস্ট্রাকটর এর আরগুমেন্ট হিসেবে কেবল একই ক্লাসের রেফারেন্স নিতে পারে এবং যখন নির্দিষ্ট কপি কনস্ট্রাকটর না থাকে কম্পাইলার অবজেক্টকে তার নিজস্ব কপি কনস্ট্রাকটর সরবরাহ করে।

নিম্নে কপি কনস্ট্রাক্টর (Copy constructor) ব্যবহার করে একটি প্রোগ্রাম দেয়া হল।

```
#include<iostream.h>
#include<conio.h>
class Student
{
private:
char *Name;
int Roll;
float Marks;
public:
Student(Student &S);
Student(char*,int,float);
void Show();
};
```

```
void Student::Student(char *N,int R,float M)
{
Name=N;
Roll=R;
Marks=M;
}
Student::Student(Student &S)
{
Name=S.Name;
Roll=S.Roll;
Marks=S.Marks;
}
void Student::Show()
{
cout<<"Your Name is:"<<Name<<endl;
cout<<"Your Roll is:"<<Roll<<endl;
cout<<"Your Marks is:"<<Marks<<endl;
}
void main()
{
clrscr();
Student S1("Mahbub",96002,88.5);
cout<<"Record of S1:"<<endl;
S1.Show();
Student S2=S1;
cout<<"Record of S2:"<<endl;
S2.Show();
getch();
}
```

ডেস্ট্রাক্টর (Destructor) ঃ

Destructor অর্থ ধ্বংসকারী। কনস্ট্রাক্টর যেমন কোন Object ঘোষণার সময় তার মেম্বারগুলোর জন্য প্রয়োজনীয় মেমোরি বরাদ্দ করে, ডেস্টাক্টর ফাংশন সেই দখলকৃত মেমোরি খালি করে দেয়।

Constructor function যেমন তার Class এর নাম ব্যবহার করে তেমনি destructor function-ও একই নাম ব্যবহার করে শুধুমাত্র শুরুতে tilde (~) অক্ষর বসিয়ে function ঘোষণা ও বর্ণনা করতে হয়। destructor ফাংশন কখনও কোন Value Return করে না। কিংবা কোন arguments গ্রহণ করে না। কোন Class এ Constructor ফাংশন থাকলে destructor ফাংশন থাকবেই।

উদাহরণ ঃ

```
class Student
{
        long Roll;
        char Name [40];
protected :
        float Marks;
public :
        Student ( );   // Constructor Defined
        ~ Student ( ); // Destructor Defined
        void Display ( );
};
Student : : Student ( ) // Constructor Defined
   {
   // Constructor Body
   }
~ Student ( ) //Destructor Defined
   {
   // Destructor Body
   }
}; // End of class Decleration
```

ডেস্ট্রাক্টর ফাংশনের বৈশিষ্ট্য ঃ

- একটি ক্লাসে মাত্র একটি destructor function থাকবে।
- ক্লাসের public অংশে ঘোষণা হয় এবং ক্লাসের নাম অনুযায়ী হয়।
- Destructor function কোন Value Return করে না। void টাইপ ও লিখা যায় না।
- const বা static হিসাবে declare করা যায় না।
- Arguments বা parameter ব্যবহার করা যায় না।
- Overload করা যায় না।
- Default Argumerts থাকতে পারে বা Virtual function হতে পারে।

ফ্রেন্ড ফাংশন (friend function) ঃ ফ্রেন্ড ফাংশন এক বিশেষ ধরনের নন-মেম্বার ফাংশন যা কোন ক্লাসের প্রাইভেট, পাবলিক এবং প্রটেকটেড মেম্বার ভেরিয়েবল বা মেম্বার ফাংশন এক্সেস করতে পারে। তবে সাধারণত ক্লাসের প্রাইভেট মেম্বার এক্সেস করার জন্য ফ্রেন্ড ফাংশন ব্যবহৃত হয়। ফ্রেন্ড ফাংশনকে ক্লাসের প্রাইভেট, পাবলিক, প্রটেক্টেড যে কোন অংশে ডিক্লেয়ার করা যায়। ফ্রেন্ড ফাংশন ডিক্লারেশনের ক্ষেত্রে ReturnType এর পূর্বে friend কীওয়ার্ড ব্যবহৃত হয়। ফ্রেন্ড ফাংশন ডিক্লারেশনের ফরম্যাট নিম্নরূপ ঃ

```
friend ReturnType FunctionName (ArgumentList);
```

উদাহরণ ঃ

```
class Student {
private :
        long Roll;
public :
        float     Marks;
        friend void GetData ( ) // Inside Definition
{
// Function Body
}
friend void Display (Student S);
};
void Display (Student S) // Outside Definition
{
// Function Body
}
```

উল্লেখ্য যে ফ্রেন্ড ফাংশনকে ক্লাসের বাইরে ডিক্লেয়ার করলে ফাংশনের রিটার্ন টাইপের পূর্বে 'friend' কীওয়ার্ড এবং স্কোপ রেজুলুশন অপারেটর (: :) ব্যবহারের প্রয়োজন হয় না। নিম্নে ফ্রেন্ড ফাংশনের জন্য একটি প্রোগ্রাম লিখা হল।

```
#include<iostream.h>
#include<conio.h>
class Student
{
private:
char Name[30];
int Roll;
float Marks;
void Input();
friend void Show(Student S);
};
void Student::Input()
{
cout<<"Enter Your Name:";
cin>>Name;
cout<<"Enter Your Roll:";
cin>>Roll;
cout<<"Enter Your Marks:";
cin>>Marks;
}
```

```
inline void Show(Student S)
{
S.Input();
cout<<"Your Name is:"<<S.Name;
cout<<endl<<"Your Roll is:"<<S.Roll;
cout<<endl<<"Your Mark is:"<<S.Marks;
}
void main()
{
clrscr();
Student Obj;
Show(Obj);
getch();
}
```

৪.৮ সি++ ভাষায় ইনহেরিটেন্স (Inheritance in C++) ঃ

ইনহেরিটেন্স (Inheritance) ঃ পুনঃব্যবহারযোগ্যতা (Reusability) অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং-এর একটি গুরুত্বপূর্ণ বৈশিষ্ট্য। একই প্রোগ্রামে কিছু অংশ বারবার লেখার চেয়ে একবার লিখে তা পুনঃব্যবহার করা সুবিধাজনক। এতে শুধু সময় ও খরচ কমে না, বরং ঝামেলা কমে এবং প্রোগ্রামের সৌন্দর্য্য ও গ্রহণযোগ্যতা বৃদ্ধি পায়। পূর্বে তৈরি কোন ক্লাস যা নির্ভুল, পরীক্ষিত এবং এক বা একাধিকবার ব্যবহৃত, তা নতুন করে তৈরি করার চেয়ে পুনরায় ব্যবহার করা নিশ্চয়ই সুবিধাজনক। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং হিসেবে সি++ পুরোপুরি এই সুবিধা দেয়। মূলত সি++ ভাষায় ক্লাসের ইনহেরিটেন্সের মাধ্যমে পুনঃব্যবহার প্রক্রিয়া বাস্তবায়ন করা হয়। সি++ এ একই ক্লাস বিভিন্নভাবে পুনঃব্যবহার করা যায়। যদি কোন প্রোগ্রামার তার প্রোগ্রামের জন্য কোন ক্লাস বর্ণনা করে তা সংরক্ষণ করেন তবে অন্যান্য প্রোগ্রামারগণও তাদের প্রোগ্রামে এই ক্লাস ব্যবহার করতে পারেন, আবার এই ক্লাসের বৈশিষ্ট্য সম্পূর্ণ বা আংশিকভাবে ব্যবহার করে নতুন কোন ক্লাস তৈরি করতে পারেন; এভাবে পুরাতন কোন ক্লাস থেকে নতুন ক্লাস তৈরি করার কৌশলকে ইনহেরিটেন্স বলা হয়। ইনহেরিটেন্স প্রক্রিয়ায় পুরাতন এক বা একাধিক ক্লাস থেকে নতুন এক বা একাধিক ক্লাস তৈরি করা হয়। সেক্ষেত্রে পুরাতন ক্লাস বা ক্লাসকে বেস ক্লাস এবং নতুন ক্লাস বা ক্লাসসমূহকে ডিরাইভড (ইনহেরিটেড) ক্লাস বলা হয়। উল্লেখ্য, প্রথম লেভেলের বেস ক্লাসকে ডাইরেক্ট বেস ক্লাস এবং পরবর্তী যে কোন লেভেলের বেস ক্লাসকে ইনডাইরেক্ট বেস ক্লাস বলা হয়। ডিরাইভড ক্লাস বেস ক্লাস হতে সম্পূর্ণ কিংবা এক বা একাধিক বৈশিষ্ট্য ইনহেরিট করে। সাধারণত বেস ক্লাসের জন্য কোন অবজেক্ট ঘোষণা করা হয় না। এরূপ বেস ক্লাসকে অ্যাবস্ট্রাক্ট ক্লাস (Abstract class) বলা হয়।

ডিরাইভড ক্লাস গঠন ঃ ইনহেরিটেন্স প্রক্রিয়ায় কোন বেস ক্লাস (Base class) হতে ডিরাইভড ক্লাস (Derived class) গঠনের জন্য class কীওয়ার্ড-এর পরে ডিরাইভড ক্লাসের নাম দিতে হয়, অতঃপর একটি কোলন দিয়ে ডিরাইভড ক্লাসে বেস ক্লাসের এক্সেসিবিলিটি মোড (যাকে ইনহেরিটেন্স মোড বলা হয়) এবং ডিরাইভড ক্লাসের নাম উল্লেখ করতে হয়। অতঃপর ওপেনিং দ্বিতীয় বন্ধনী দ্বারা ডিরাইভড ক্লাস শুরু করতে হয় এবং ক্লোজিং দ্বিতীয় বন্ধনী এবং সেমিকোলন দ্বারা তা শেষ করতে হয়। ওপেনিং এবং ক্লোজিং দ্বিতীয় বন্ধনীর মধ্যে সাধারণ ক্লাস ঘোষণার মত ডিরাইভড ক্লাসের নিজস্ব মেম্বার ভেরিয়েবল এবং মেম্বার ফাংশন ঘোষণা করা হয়। অতঃপর ডিরাইভড ক্লাসের নিজস্ব মেম্বার ফাংশন বর্ণনা/(ডিফাইন) করা হয় এবং তা মেম্বার ভেরিয়েবলগুলোর জন্য মান নির্ধারণ করা হয়। এই বর্ণনার মাঝে নিয়মানুযায়ী সুবিধাজনক স্থানে বেস ক্লাসের ইনহেরিটেড মেম্বার এক্সেস করা হয়। বেস ক্লাসের ইনহেরিটেন্স মোড একটি অপশনাল

অংশ। ইনহেরিটেন্স মোড বলতে প্রাইভেট (private) কিংবা পাবলিক (public) এই দুইয়ের যে কোন একটি বুঝায়। তবে ইনহেরিটেন্স মোড উল্লেখ না করলে ডিফল্ট হিসেবে প্রাইভেট মোড ধরে নেয়া হয়। বেস ক্লাসের পাবলিক এবং প্রটেকটেড মেম্বারসমূহ ডিরাইভড ক্লাসে কোন মোডে ইনহেরিট হবে তা ইনহেরিটেন্স মোড দিয়ে দেখান হয়। উল্লেখ্য, বেস ক্লাসের প্রাইভেট মেম্বারসমূহ কখনই ডিরাইভড ক্লাসে ইনহেরিট করা যায় না। একাধিক বেস ক্লাস থাকলে প্রতিটি বেস ক্লাসের মধ্যে কমা দিতে হয় এবং প্রত্যেকের জন্য আলাদা ইনহেরিটেন্স মোড উল্লেখ করতে হয়। নিম্নে একটি বেস ক্লাস থেকে ডিরাইভড ক্লাস গঠনের ফরম্যাট দেয়া হল। এখানে BaseClass কে বেস ক্লাস এবং DerivedClass কে ডিরাইভড ক্লাস হিসেবে ধরা হয়েছে।

```
class BaseClass
{
private :
        // private members; Not inheritable
protected :
        // protected members : Inheritable
public :
        // public members : Inheritable
}; // end of base class
class DerivedClass : BaseClassAccessibilityMode BaseClass
{
private :
        // private members : Not further inheritable
public :
        // public members : Ready for further inheritance
};      // end of derived class
```

ইনহেরিটেন্স-এর বিভিন্ন মোড (Modes of Inheritance) ঃ

কোন ক্লাসে সাধারণত প্রাইভেট, পাবলিক এবং প্রটেকটেড এই তিন ধরনের মেম্বার ভেরিয়েবল এবং মেম্বার ফাংশন থাকে। বেস ক্লাসের মেম্বারগুলোকে ডিরাইভড ক্লাসে প্রাইভেট (private) এবং পাবলিক (public) এই এই দুইটি আলাদা মোডে ইনহেরিট করা যায়। এগুলোকে যথাক্রমে প্রাইভেট ইনহেরিটেন্স মোড এবং পাবলিক ইনহেরিটেন্স মোড বলা হয়। প্রত্যেক মোড ইনহেরিটেন্সের ফলাফল আলাদা। নিম্নের ছকে প্রাইভেট এবং পাবলিক মোডে ইনহেরিটেন্সের জন্য ডিরাইভড ক্লাসে বেস ক্লাসের প্রাইভেট, পাবলিক এবং প্রটেকটেড মেম্বারের ভিজিবিলিটি মোড দেখান হল। উল্লেখ্য, বেস ক্লাসের প্রাইভেট মেম্বারসমূহ কখনই ডিরাইভড ক্লাসে ইনহেরিট করা যায় না।

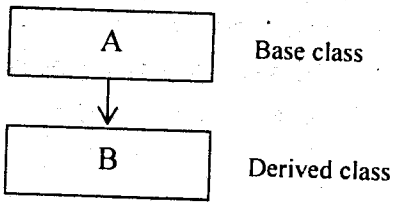
বেস ক্লাস মেম্বার ভিজিবিলিটি মোড	ডিরাইভড ক্লাস মেম্বার ভিজিবিলিটি মোড	
	প্রাইভেট ইনহেরিটেন্স	পাবলিক ইনহেরিটেন্স
প্রাইভেট	নট ইনহেরিটেবল	নট ইনহেরিটেবল
প্রটেকটেড	প্রাইভেট	প্রটেকটেড
পাবলিক	প্রাইভেট	পাবলিক

টেবিল ঃ ইনহেরিটেড মেম্বারসমূহের ভিজিবিলিটি মোড

ইনহেরিটেন্স মোড উল্লেখ না করলে ডিফল্ট হিসেবে প্রাইভেট মোড ধরে নেয়া হয়। বেস ক্লাসের পাবলিক এবং প্রটেকটেড মেম্বারসমূহ ডিরাইভড ক্লাসে কোন মোডে ইনহেরিট হবে তা ইনহেরিটেন্স মোড দিয়ে দেখান হয়।

বিভিন্ন প্রকার ইনহেরিটেন্স (Different Types of Inheritance) ঃ

সিংগল (single) ইনহেরিটেন্স ঃ যখন কোন ডিরাইভড (Derived) ক্লাস একটিমাত্র বেস (Base) ক্লাস থেকে এক বা একাধিক বৈশিষ্ট্য ইনহেরিট করে তখন এই প্রক্রিয়াকে সিঙ্গেল ইনহেরিটেন্স বলে।



চিত্র ঃ সিঙ্গেল ইনহেরিটেন্স

ফরম্যাট ঃ

এই প্রক্রিয়ায় ক্লাস ব্যবহারের ফরম্যাট দেখানো হল ঃ

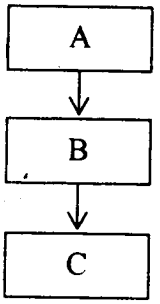
```
BaseClass{
private :
        //private members;
public :
        //private members;
};
DerivedClass : public BaseClass
{
------------  //body
------------  //
};
```

প্রোগ্রাম ঃ

```
#include<iostream.h>
#include<conio.h>
class Result{
public:
int a,b,result ;
};
class add:public Result
{
public:
void Show();
};
void add::Show()
{
cout<<"Enter the value of a,b:";
cin>>a>>b;
```

```
result=a+b;
cout<<"a+b is="<<result;
}
void main()
{
clrscr();
add A;
A.Show();
getch();
}
```

মাল্টিলেভেল ইনহেরিটেন্স (Multilevel Inheritance) : যখন কোন derived class অপর কোন derived class এর এক বা একাধিক বৈশিষ্ট্য inherit করে তখন এই প্রক্রিয়াকে মাল্টিলেভেল ইনহেরিটেন্স প্রক্রিয়া বলে।



Base Class

Intermediate Base Class

Derived Class

চিত্র ঃ মাল্টিলেভেল ইনহেরিটেন্স

ফরম্যাট ঃ

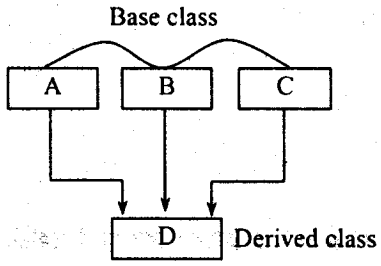
```
class A
{
// ----------
};
class B : public A
{
//----------
};
class C: public B
{
// -----------
};
```

উপরের ঘোষণা থেকে আমরা দেখি C ক্লাসটি B ক্লাসের বৈশিষ্ট্য ইনহেরিট করে এবং B ক্লাসটি A ক্লাসের বৈশিষ্ট্য ইনহেরিট করে।

প্রোগ্রাম ঃ

```
#include<iostream.h>
#include<conio.h>
class A
{
public:
int a,b;
};
class B:public A
{
public:
int result;
};
class add:public B
{
public:
void Show();
};
void add::Show()
{
cout<<"Enter the value of a,b:";
cin>>a>>b;
result=a+b;
cout<<"a+b is="<<result;
}
void main()
{
clrscr();
add obj;
obj.Show();
getch();
}
```

মাল্টিপল ইনহেরিটেন্স (Multiple Inheritance) ঃ কোন ডিরাইভড ক্লাস দুই বা ততোধিক বেস ক্লাস থেকে বৈশিষ্ট্য ইনহেরিট করলে এই প্রক্রিয়াকে মাল্টিপল ইনহেরিটেন্স বলা হয়।



চিত্র ঃ মাল্টিপল ইনহেরিটেন্স

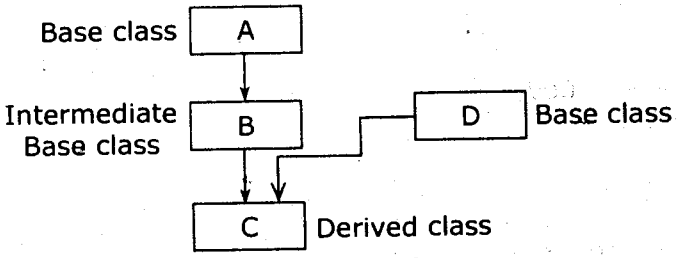
ফরম্যাট ঃ

```
class D : public A, public B, public C
{
--------------
-------------- / Body
--------------
};
```

প্রোগ্রাম ঃ

```
#include<iostream.h>
#include<conio.h>
class A
{
public:
int a,b;
};
class B
{
public:
int result;
};
class add:public A,public B
{
public:
void Show();
};
void add::Show()
{
cout<<"Enter the value of a,b:";
cin>>a>>b;
result=a+b;
cout<<"a+b is="<<result;
}
void main()
{
clrscr();
add obj;
obj.Show();
getch();
}
```

হাইব্রিড ইনহেরিটেন্স ঃ যে পদ্ধতিতে দুই বা ততোধিক পদ্ধতির ইনহেরিটেন্স একত্রে ব্যবহার করার প্রয়োজন হয় তাকে হাইব্রিড ইনহেরিটেন্স বলে।



চিত্র ঃ হাইব্রিড ইনহেরিটেন্স

ফরম্যাট ঃ

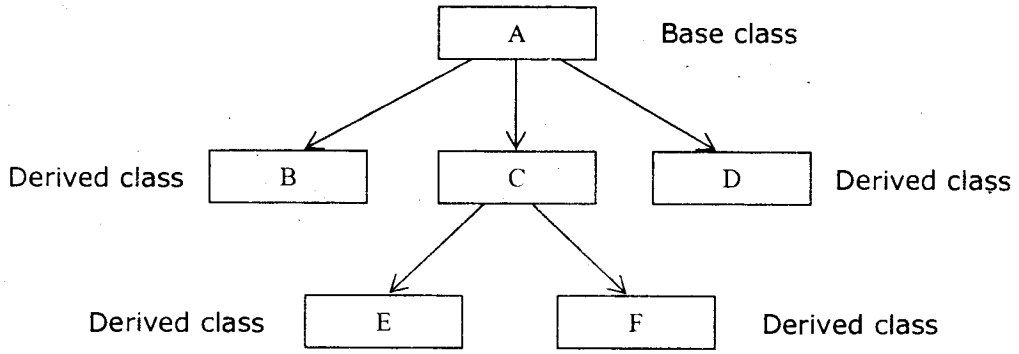
```
class A
 {
      // Body
 }
class B : public A
 {
      // Body
 }
class D
 {
      // Body
 }
class C : public B, public D
 {
      // Body
 }
```

প্রোগ্রাম ঃ

```
#include<iostream.h>
#include<conio.h>
class Student
{
public:
char Name[30];
int Roll;
void GetData();
};
class Exam:public Student
{
float Marks;
```

```
public:
void Input();
void Show();
};
class Sessional
{
private:
float Sessional_Marks;
public:
void GetSessional();
};
void Student::GetData()
{
cout<<"Enter Your Name:";
cin>>Name;
cout<<endl<<"Your Name is:"<<Name;
cout<<endl<<"Enter Your Roll:";
cin>>Roll;
cout<<endl<<'Your Roll is:"<<Roll;
}
void Exam::Input()
{
cout<<endl<<"Enter Your Marks:";
cin>>Marks;
cout<<endl<<"Your Mark is:"<<Marks;
}
void Exam::Show()
{
GetData();
Input();
}
void Sessional::GetSessional()
{
cout<<endl<<"Enter Your Sessional Marks:";
cin>>Sessional_Marks;
cout<<endl<<"Your Sessional Mark is:"<<Sessional_Marks;
}
void main()
{
clrscr();
Exam Obj;
Obj.Show();
Sessional S;
S.GetSessional();
getch();
}
```

হায়ারারকিক্যাল ইনহেরিটেন্স ঃ hierarchical inheritance হল মাল্টিপল ইনহেরিটেন্সের বিপরীত অবস্থান। অর্থাৎ যে পদ্ধতিতে একটি মাত্র বেস (Base) ক্লাস হতে দুই বা ততোধিক ডিরাইভড ক্লাস গঠন করা হয় তাকে হায়ারারকিক্যাল ইনহেরিটেন্স বলে।



চিত্র ঃ হায়ারারকিক্যাল ইনহেরিটেন্স

ফরম্যাট ঃ

```
BaseClass {
--------------
--------------
};
De...  : BaseClass{
---------
--------------
};
Derived2 : BaseClass{
--------------
--------------
};
DerivedN : BaseClass{
--------------
--------------
};
```

অনুশীলনী-৪

অতি সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। সি++ কে অবজেক্ট অরিয়েন্টেড ল্যাংগুয়েজ বলা হয় কেন?

উত্তরঃ যেহেতু সি++ ভাষায় অবজেক্ট অরিয়েন্টেড ল্যাংগুয়েজের সকল গুরুত্বপূর্ণ বৈশিষ্ট্যসমূহ (যেমন- ক্লাস, ইনহেরিটেন্স, পলিমরফিজম, ডাটা অ্যাবস্ট্রাকশন, এনক্যাপসুলেশন) বিদ্যমান তাই সি++ কে অবজেক্ট অরিয়েন্টেড ল্যাংগুয়েজ বলা হয়।

২। সর্বপ্রথম কে এবং কখন সি++ ল্যাংগুয়েজ ডেভেলপ করেন?

উত্তরঃ সর্বপ্রথম ১৯৮৩ সালে বেল ল্যাবরেটরির গবেষক জর্ন স্ট্রাউস্ট্রাপ (Bjarne stroustrup) সি++ ল্যাংগুয়েজ ডেভেলেপ করেন।

৩। কয়েকটি C ++ কম্পাইলারের নাম লিখ।

উত্তরঃ কয়েকটি বহুল ব্যবহৃত C++ কম্পাইলারের নাম হল ANSI C ++, Borland C++, Microsoft C++, Turbo C++, GNU C++ ইত্যাদি।

৪। ডাটা স্ট্রিম বলতে কি বুঝ?

উত্তরঃ কোন প্রোগ্রামে কীবোর্ড কিংবা ফাইল থেকে ডাটা নিয়ে ভেরিয়েবলে রাখা কিংবা ভেরিয়েবল থেকে ডাটা নিয়ে মনিটর কিংবা ফাইলে রাখা মানে হল ডাটাকে এক জায়গা থেকে অন্য জায়গায় পাঠানো। সহজ ভাষায় ডাটা এক জায়গা থেকে অন্য জায়গায় পাঠানোকে তথা ডাটার প্রবাহকে data stream বলে।

৫। কনসোল I/O কি? ইহা কত প্রকার ও কী কী?

উত্তরঃ C++ প্রোগ্রামে ইনপুট আউটপুট অপারেশন সম্পন্ন করার জন্য যে সকল statement ব্যবহার করা হয় তাদেরকে কনসোল I/O বলে। ইহা দুই প্রকার। যথা ঃ ১। Formatted I/O এবং ২। Unformatted I/O.

৬। ফরম্যাটেড I/O অপারেশনের প্রয়োজনীয়তা কী?

উত্তরঃ প্রোগ্রামিং এর মাধ্যমে কোন সমস্যা সমাধানের জন্য প্রোগ্রামের একটি অতীব গুরুত্বপূর্ণ বিষয় হচ্ছে input output operation. ইনপুট অপারেশন ও আউটপুট অপারেশন ছাড়া প্রোগ্রামে কোন ডাটা ইনপুট নেয়া যায় না কিংবা প্রসেসকৃত result আউটপুট হিসেবে পাঠানো যায়না। আর C++ এ এই সকল operation কতগুলো formatted statement এর মাধ্যমে সম্পন্ন করার পদ্ধতিকে formatted I/O operation বলে। তাই বলা যায় C++ এ formatted I/O operation এর প্রয়োজনীয়তা অপরিসীম।

৭। ফরম্যাটেড I/O র কাজ কয়ভাবে ও কী কী ভাবে করা যায়?

উত্তরঃ C++ এর i/o stream লাইব্রেরী ব্যবহার করে formatted i/o'র কাজ তিন ভাবে করা যায়। যথা ঃ (১) manipulator ব্যবহার করে (২) ios class এর মেম্বার ফাংশন এবং flag (৩) user defined output functions ব্যবহার করে।

৮। Unformatted input operation এ ব্যবহৃত ফাংশন সমূহের নাম লিখ।

উত্তরঃ Unformated input operation এর জন্য সাধারণতঃ cin, get ( ), getline ( ), read ( ), ignore ( ), peek ( ), gcount ( ), putback ( ), sync ( ) ইত্যাদি function ব্যবহৃত হয়।

৯। Unformatted Output Operation এ ব্যবহৃত ফাংশনসমূহের নাম লিখ।

উত্তরঃ Unformatted output operation বলতে সাধারণতঃ cout, put ( ), write ( ), tellp ( ) ইত্যাদি বুঝায়।

১০। ডট অপারেটর বলতে কী বুঝ?

উত্তরঃ সাধারণতঃ iostream stream এর সাথে unformatted ও formatted input output operation এর সংযোগ ঘটাতে যে ধরনের Operator (.) ব্যবহার করা হয় তাকে ডট অপারেটর (.) বলা হয়।

১১। ক্যাসকেডিং আই/ ও অপারেটর বলতে কী বোঝায়?

উত্তরঃ একই স্টেটমেন্টে একাধিক ইনপুট অপারেটর (>>) কিংবা আউটপুট অপারেটর ব্যবহৃত হলে তাকে ক্যাসকেডিং আই/ও অপারেটর বলে।

যেমন ঃ cin >> a >>b;

cout << a <<b;

১২। endl, ends, setw, ws অপারেটর-এর ব্যবহার লেখ।

উত্তরঃ endl ঃ আউটপুটকে নতুন লাইনের শুরুতে প্রদর্শনের জন্য endl অপারেটর ব্যবহৃত হয়।

ends ঃ কোন স্ট্রিং বা সংখ্যার শেষে একটি নাল ক্যারেক্টার ('\০') সংযুক্ত করার জন্য ends অপারেটর ব্যবহৃত হয়।

setw ঃ আউটপুট ভ্যালু প্রদর্শনের ক্ষেত্রে প্রয়োজনীয় ফিল্ড সাইজ স্পেসিফাই করার জন্য setw ব্যবহৃত হয়।

ws ঃ কী বোর্ড থেকে কোন মান ইনপুট নেয়ার সময় প্রথম শব্দ বা ক্যারেক্টারের শুরুতেই কোন ফাঁকা স্থান পড়লে তা বাদ দেয়ার জন্য ws অপারেটর ব্যবহৃত হয়।

১৩। সি++ এ width ( ) এবং precission ( ) ফাংশন দুটির কাজ লেখ।

উত্তরঃ width ( ) ঃ অউটপুট ভ্যালু প্রদর্শনের জন্য প্রয়োজনীয় ফিল্ড সাইজ স্পেসিফাই করার জন্য width () ফাংশন ব্যবহৃত হয়।

precission ( ) ঃ ফ্লোটিং পয়েন্ট টাইপের মান প্রদর্শনের ক্ষেত্রে দশমিকের পর কত ডিজিট পর্যন্ত প্রদর্শিত হবে তা নিরূপণের জন্য precission ( ) ফাংশন ব্যবহৃত হয়।

১৪। সি++ এ ব্যবহৃত ৫টি কীওয়ার্ডের নাম লেখ যেগুলো সি ভাষার অন্তর্ভুক্ত নয়।

উত্তরঃ শুধুমাত্র সি++ এ ব্যবহৃত ৫টি কীওয়ার্ড হল ঃ catch, class, new, delete, private, public, protected ইত্যাদি।

১৫। সি++ এ ব্যবহৃত কয়েকটি স্পেশাল অপারেটরের নাম লেখ।

উত্তরঃ সি++ এ ব্যবহৃত কয়েকটি স্পেশাল অপারেটর হল ঃ ইনসার্শন অপারেটর (<<), এক্সট্রাকশন অপারেটর (>>), স্কোপ রেজুলেশন অপারেটর (: :), new, delete, endl, ends, ws ইত্যাদি।

১৬। সি++ এ ব্যবহৃত মেমোরি ম্যানেজমেন্ট অপারেটরসমূহের নাম লেখ।

উত্তরঃ সি++ এ ব্যবহৃত মেমোরি ম্যানেজমেন্ট অপারেটরসমূহ হচ্ছে ঃ new এবং delete.

১৭। (: :) অপারেটরের কাজ কী?

উত্তরঃ ফাংশন কোন ক্লাসের অন্তর্ভুক্ত তা নির্দেশ করার জন্য (: :) অপারেটর ব্যবহৃত হয়।

১৮। C++ এ ব্যবহৃত বিভিন্ন ধরনের function এর নাম লিখ।

উত্তরঃ C++ এ ব্যবহৃত বিভিন্ন ধরনের function গুলো নিম্নরূপ । যথা ঃ
রিকার্সিভ ফাংশন, স্ট্যাটিক ফাংশন, ইনলাইন ফাংশন, এক্সটার্নাল ফাংশন, ফ্রেন্ড ফাংশন এবং ভার্চুয়াল ফাংশন ইত্যাদি।

১৯। রিকার্সন বলতে কী বুঝায়?

উত্তরঃ আমরা জানি, একটি ফাংশন অন্য কোন ফাংশনকে যে কোন সংখ্যক বার কল করতে পারে। আবার কোন ফাংশন নিজেও নিজেকে কল করতে পারে। যখন কোন ফাংশন নিজেই নিজেকে কল করে তখন তাকে রিকার্সিভ ফাংশন বলা হয় এবং এই প্রক্রিয়াকে রিকার্সন (Recursion) বলা হয়।

২০। রিকার্সিভ ফাংশনের সুবিধা লিখ।

উত্তরঃ রিকার্সিভ ফাংশনের সুবিধা হল এতে প্রোগ্রামের জটিলতা অনেক কমে যায়, এতে অনেক কম সংখ্যক ভেরিয়েবলের প্রয়োজন হয় এবং প্রোগ্রাম সহজ ও সুন্দর হয়।

২১। ইনলাইন ফাংশন কাকে বলে?

উত্তরঃ ফাংশন ব্যবহারের সময় ব্যবহারীকারী ফাংশনের পরবর্তী স্টেটমেন্টের রিটার্ন অ্যাড্রেস সেভ করা, ফাংশনের ফিজিক্যাল অ্যাড্রেস গণণা করা, আরগুমেন্টগুলোর একচুয়াল মান ফেরৎ পাঠানো এবং সবশেষে ব্যবহারকারী ফাংশনের পরবর্তী স্টেটমেন্টের রিটার্ন অ্যাড্রেস গণণা করে তার কার্যক্রম শুরু করা প্রভৃতি কাজের জন্য অতিরিক্ত সময় ব্যয় রোধ করার জন্য C/C++প্রোগ্রামে ম্যাক্রো ব্যবহার করা হয়। তবে ম্যাক্রো ব্যবহারের একটি অসুবিধা হল, ম্যাক্রো কোন ফাংশন নয়, তাই কম্পাইলার ম্যাক্রো কম্পাইলিং এর সময় তার ভুল বা অসঙ্গতি নিরীক্ষা করে না। C++এ এরূপ ছোট কোন প্রোগ্রামে বার বার ব্যবহারের ক্ষেত্রে একটি বিশেষ ধরনের ফাংশন ব্যাপকভাবে ব্যবহৃত হয় যা ইনলাইন (inline) ফাংশন নামে পরিচিত।

২২। ফাংশন আরগুমেন্ট বলতে কী বুঝায়?

উত্তরঃ ফাংশন calling এর সময় এর মধ্যে যে মান বা ভেরিয়েবল ব্যবহার করা হয় তাকে ফাংশন আরগুমেন্ট বলে।

২৩। Constant member function কী?

উত্তরঃ Constant member function হলো এমন এক ধরনের number function যা কখনো তার class এর কোন member variable এর ডাটা পরিবর্তন করবেনা। একে নিম্নরূপে declare করা যায়ঃ

```
ReturnType FunctionName (ParameterList) const
{ - - - - }
```

২৪। ফাংশন প্রটোটাইপ কিভাবে গঠিত হয়?

উত্তরঃ সাধারণতঃ রিটার্ন টাইপ, ফাংশন, আইডেন্টিফায়ার এবং প্যারামিটারের ধরণ মিলে ফাংশন প্রটোটাইপ গঠিত হয়।

২৫। ফাংশন ওভারলোডিং বলতে কী বুঝায়?

উত্তরঃ দুই বা ততোধিক ফাংশনের একই নাম হলে তাকে ফাংশন ওভারলোডিং বলে।

২৬। মেম্বার ফাংশন ও মেম্বার ভেরিয়েবল কাকে বলে?

উত্তরঃ ক্লাসের অভ্যন্তরে ডিক্লেয়ারকৃত ফাংশনসমূহকে মেম্বার ফাংশন বলে। আর ক্লাসের অভ্যন্তরে ডিক্লেয়ারকৃত ভেরিয়েবলসমূহকে মেম্বার ভেরিয়েবল বলে।

২৭। মেম্বার ফাংশনকে কয়ভাবে define করা যায় ও কী কী?

উত্তরঃ মেম্বার ফাংশনকে দুইভাবে define করা যায়। যথা ঃ

outside the class definition এবং inside the class definition.

২৮। নেস্টেড মেম্বার ফাংশন বলতে কী বুঝায়?

উত্তরঃ একই ক্লাসের একই জাতীয় কোন মেম্বার ফাংশন অন্য কোন member function কে call করতে পারে। member function কর্তৃক member function কে call করার এই প্রক্রিয়াকে nested member function বলে।

২৯। স্ট্যাটিক মেম্বার ফাংশন কাকে বলে?

উত্তরঃ কোন member function এর returndatatype এর পূর্বে কোন static keyword ব্যবহার করলে সেই ফাংশনটিকে static member function বলে।

৩০। অপারেটর ওভারলোডিং কাকে বলে?

উত্তরঃ কোন অপারেটরের বিল্ট-ইন যে কাজ করার ক্ষমতা থাকে তার চেয়ে বাড়তি কিছু কাজ করার ক্ষমতা তৈরির প্রক্রিয়াকে অপারেটর ওভারলোডিং বলে। যেমন ঃ প্লাস অপারেটর (+) দুটি সংখ্যা যোগ করার কাজে ব্যবহৃত হয়। আবার অপারেটর ওভারলোডিং প্রক্রিয়ায় দুটি স্ট্রিং এর যোগফল হিসেবে অপর একটি স্ট্রিং আউটপুট প্রদান করতে পারে।

৩১। কনস্ট্রাক্টর কাকে বলে?

উত্তরঃ কনস্ট্রাক্টর হল এমন এক বিশেষ ধরনের ফাংশন যা কোন অবজেক্ট ঘোষণার সময় তার মান নির্ধারণ করে এবং মেমোরি স্পেস বরাদ্দ করে।

৩২। কপি কনস্ট্রাক্টর বলতে কী বোঝায়?

উত্তরঃ কোন কনস্ট্রাক্টর ফাংশনের আরগুমেন্ট হিসাবে একই ক্লাসের কোন রেফারেন্স অ্যাড্রেস ব্যবহৃত হলে তাকে কপি কনস্ট্রাক্টর বলে।

৩৩। মাল্টিপল কনস্ট্রাক্টর কাকে বলে?

উত্তরঃ কোন ক্লাসে একই সাথে দুই বা ততোধিক কনস্ট্রাক্টর ব্যবহার করা হলে তাকে মাল্টিপল কনস্ট্রাক্টর বলে।

৩৪। কপি কনস্ট্রাক্টরের ব্যবহার লেখ।

উত্তরঃ একটি নির্দিষ্ট অবজেক্ট থেকে এক বা একাধিক অবজেক্টের মান নির্ধারণের জন্য কপি কনস্ট্রাক্টর ব্যবহৃত হয়।

৩৫। ডেস্ট্রাক্টর বলতে কী বোঝায়?

উত্তরঃ ডেস্ট্রাক্টর হচ্ছে এমন একটি ফাংশন যা কনস্ট্রাক্টর কর্তৃক বরাদ্দকৃত মেমোরি স্পেস ধ্বংস করে বা খালি করে দেয়।

৩৬। ক্লাসে public ও private কীওয়ার্ড ব্যবহার করা হয় কেন?

উত্তরঃ ক্লাসে public কীওয়ার্ড ব্যবহার করে ঐ অংশে মেম্বার ভেরিয়েবল ও মেম্বার ফাংশন ডিক্লেয়ার করলে যে কোন ক্লাস ঐ ভেরিয়েবল ও ফাংশনকে ব্যবহার করতে পারবে। অন্যদিকে privete অংশে ডিক্লেয়ারকৃত ভেরিয়েবল ও ফাংশনসমূহ কেবলমাত্র সংশ্লিষ্ট ক্লাসের জন্যই ব্যবহৃত হবে। অন্য কোন ক্লাস সেগুলো এক্সেস করার সুযোগ পাবে না। এতে সিস্টেমের নিরাপত্তা বাড়বে।

৩৭। ক্লাসে protect কীওয়ার্ড কেন ব্যবহার করা হয়?

উত্তরঃ ক্লাসের protect অংশে ডিক্লেয়ারকৃত ভেরিয়েবল ও ফাংশনকে সংশ্লিষ্ট ক্লাস ও বিশেষ প্রয়োজনে বিশেষ ব্যবস্থায় অন্য কোন ক্লাস ও এক্সেস করতে পারে। সুতরাং protect কীওয়ার্ড ব্যবহার করেও সিস্টেমের নিরাপত্তা বিধান করা যায়।

৩৮। Base class ও Derived class কাকে বলে?

উত্তরঃ কোন একটি ক্লাস যদি পূর্বেই ডিফাইনকৃত অন্য কোন ক্লাসের বৈশিষ্ট্য ইনহেরিট করে তবে যে ক্লাস থেকে ইনহেরিট করা হয় তাকে Base class আর যে Class বৈশিষ্ট্য ইনহেরিট করে তাকে Derived class বলে।

৩৯। মাল্টিপল ইনহেরিটেন্স কাকে বলে?

উত্তরঃ যখন কোন Derived class দুই বা ততোধিক Base class থেকে দুই বা ততোধিক বৈশিষ্ট্য inherit করে তবে সেই প্রক্রিয়াকে মাল্টিপল ইনহেরিটেন্স বলা হয়।

৪০। মাল্টিলেভেল ইনহেরিটেন্স কাকে বলে?

উত্তরঃ যখন কোন derived class অপর কোন base class এর derived class থেকে এক বা একাধিক বৈশিষ্ট্য ইনহেরিট করে তবে তাকে Multilevel inheritance বলে।

৪১। হায়ারারকিক্যাল ইনহেরিটেন্স কাকে বলে?

উত্তরঃ যখন একটি মাত্র Base class হতে দুই বা ততোধিক derived class গঠন করা হয় তবে সেই প্রক্রিয়াকে হায়ারারকিক্যাল ইনহেরিটেন্স বলে।

৪২। হাইব্রিড ইনহেরিটেন্স কাকে বলে?

উত্তরঃ যদি কোন derived class একই সাথে multiple ও multilevel উভয় উপায়ে বিভিন্ন base class থেকে দুই বা ততোধিক বৈশিষ্ট্য inherit করে তবে সেই প্রক্রিয়াকে Hybrid inheritance বলে।

৪৩। ভার্চুয়াল Base class কাকে বলে?

উত্তরঃ Multiple inheritance এর ক্ষেত্রে একাধিক derived class একই base class কে inherit করলে এবং অপর কোন derived class পূর্বোক্ত derived class গুলোকে inherit করলে শেষোক্ত derived classটি প্রথমোক্ত base class এর একাধিক copy অর্জন করে। ফলে এক ধরনের অসামঞ্জস্যতার সৃষ্টি হয়। এ ধরনের অসামঞ্জস্যতা দূরীকরণের জন্য যে ধরনের class ব্যবহৃত হয় তাকে virtual base class বলে।

৪৪। অ্যাবস্ট্রাক্ট ক্লাস কাকে বলে?

উত্তরঃ যখন কোন ক্লাসে কমপক্ষে একটি pure ভার্চুয়াল ফাংশন থাকে তখন সেই ক্লাসকে অ্যাবস্ট্রাক্ট ক্লাস বলে।

৪৫। ফাংশন ওভারলোডিংজনিত অনিশ্চিয়তা বা অস্পষ্টতা বলতে কী বোঝায়?

অথাব, কী কী কারণে ফাংশন ওভারলোডিংজনিত অনিশ্চয়তা দেখা দিতে পারে?

উত্তরঃ ফাংশন ওভারলোডিং এর সময় Type conversion, reference parameter, default argument এবং কোন ফাংশনকে কল করা হয়েছে সেটি না বোঝার কারণে যে ধরনের অস্পষ্টতা তৈরি হয় বা অনিশ্চয়তা দেখা দেয় তাই ফাংশন ওভারলোডিংজনিত অনিশ্চয়তা।

৪৬। ফাংশন ওভারলোডিংজনিত অনিশ্চয়তা কিভাবে দূর করা যায়?

উত্তরঃ ফাংশন ওভারলোডিং এর সময় জেনে বুঝে ডাটা টাইপ ডিক্লেয়ার করা, বুঝে শুনে সঠিক ফাংশনটি কল করা, ডিফল্ট আরগুমেন্ট ও রেফারেন্স প্যারামিটার সঠিকভাবে ব্যবহারের মাধ্যমেই ফাংশন ওভারলোডিংজনিত অনিশ্চয়তা দূর করা যায়।

সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। সি++ এ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর যে সকল বৈশিষ্ট্য বিদ্যমান সেগুলো উল্লেখ কর।

উত্তর সংকেত ঃ ৪.১ নং অনুচ্ছেদ দ্রষ্টব্য।

২। সি++ প্রোগ্রামের বেসিক স্ট্রাকচার বর্ণনা কর।

উত্তর সংকেত ঃ ৪.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। সি++ এর ফরম্যাটেড, আনফরম্যাটেড আই/ ও অপারেশন বর্ণনা কর।

উত্তর সংকেত ঃ ৪.৩.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। সি++ এ ব্যবহৃত নতুন অপারেটরগুলোর কাজ লেখ।

উত্তর সংকেত ঃ ৪.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। new এবং detete অপারেটরের ব্যবহার অথবা প্রয়োজনীয়তা বর্ণনা কর।

উত্তর সংকেত ঃ ৪.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। : :* এবং .* অপারেটর দুটির মাঝে পার্থক্য লেখ।

উত্তর সংকেত ঃ ৪.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৭। সি++ এ ফাংশন ব্যবহারের বিবেচ্য বিষয়গুলো কী কী?

উত্তর সংকেত ঃ ৪.৪.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৮। বৃত্তের ক্ষেত্রফল নির্ণয়ের জন্য ফাংশন ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ৪.৪.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৯। স্ট্রাকচার এবং ক্লাস এর মধ্যে পার্থক্য লেখ।

উত্তর সংকেত ঃ ৪.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১০। সি++ প্রোগ্রামে এক্সেস স্পেসিফায়ারের কাজ লেখ।

উত্তর সংকেত ঃ ৪.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১১। ক্লাসের public, private ও protected মেম্বার বলতে কী বোঝায়?

উত্তর সংকেত ঃ ৪.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১২। ক্লাস ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

অথবা, দেখাও যে, ক্লাসের pulbic member function ব্যতিত private member function কে এক্সেস করা যায় না।

উত্তর সংকেত ঃ ৪.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১৩। অপারেটর ওভারলোডিং এর সাধারণ নিয়মগুলো লেখ।

উত্তর সংকেত ঃ ৪.৬.১ নং অনুচ্ছেদ দ্রষ্টব্য।

১৪। মেম্বার ফাংশনকে কয়ভাবে ডিফাইন করা যায় ও কী কী?

উত্তর সংকেত ঃ ৪.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১৫। অপারেটর ওভারলোডিং এর সীমাবদ্ধতা লেখ।

উত্তর সংকেত ঃ ৪.৬.১ নং অনুচ্ছেদ দ্রষ্টব্য।

১৬। সি++ এ কনস্ট্রাক্টর কেন ব্যবহার করা হয়?

উত্তর সংকেত ঃ ৪.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

১৭। ফাংশন ওভারলোডিং এর শর্তাবলি কী কী?

উত্তর সংকেত ঃ ৪.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

১৮। ফাংশন ওভারলোডিং এর সীমাবদ্ধতা উল্লেখ কর।

উত্তর সংকেত ঃ ৪.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

১৯। **Copy constructor** তৈরি ও এর ব্যবহার উল্লেখ কর।

উত্তর সংকেত ঃ ৪.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

২০। ডেস্ট্রাক্টর ফাংশন কী? উদাহরণসহ বুঝিয়ে লেখ।

উত্তর সংকেত ঃ ৪.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

২১। ফ্রেন্ড ফাংশন কী? একটি ফ্রেন্ড ফাংশন তৈরি কর।

উত্তর সংকেত ঃ ৪.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

২২। হায়ারারকিক্যাল ইনহেরিটেন্স এর ফরম্যাট লেখ।

উত্তর সংকেত ঃ ৪.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

২৩। **Base class** এবং **Derived class** ডিক্লেয়ার করার **syntax** লেখ।

উত্তর সংকেত ঃ ৪.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

২৪। ইনহেরিটেন্সের শ্রেণিবিভাগ দেখাও।

উত্তর সংকেত ঃ ৪.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

২৫। হায়ারারকিক্যাল ও হাইব্রিড ইনহেরিটেন্স এর মধ্যে পার্থক্য লেখ।

উত্তর সংকেত ঃ ৪.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

২৬। **Base class** ও **Derived class** এর মধ্যে পার্থক্য কী?

উত্তর সংকেত ঃ ৪.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

২৭। সিঙ্গেল ইনহেরিটেন্স ও মাল্টিপল ইনহেরিটেন্স এর মধ্যে পার্থক্য লেখ।

উত্তর সংকেত ঃ ৪.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

➥ *রচনামূলক প্রশ্নোত্তর ঃ*

১। সি++ প্রোগ্রামিং এর **Console I/O** বা ফরম্যাটেড ও আনফরম্যাটেড **I/O** বর্ণনা কর।

উত্তর সংকেত ঃ ৪.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

২। বিভিন্ন প্রকার কন্ট্রোল স্ট্রাকচার বর্ণনা কর।

উত্তর সংকেত ঃ ৪.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। উদাহরণসহ সি++ এর বিভিন্ন প্রকার অপারেটরগুলোর বর্ণনা দাও।

উত্তর সংকেত ঃ ৪.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। ক্লাস ব্যবহার করে তিনটি সংখ্যার মধ্যে বৃহত্তম সংখ্যাটি নির্ণয়ের প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ৪.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। দ্বিঘাত সমীকরণের মূল (বাস্তব ও অবাস্তব) নির্ণয়ের একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ব্যবহারিক অংশ দ্রষ্টব্য।

৬। ক্লাস ব্যবহার করে n সংখ্যক ডাটাকে Sorting করার প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ব্যবহারিক অংশ দ্রষ্টব্য।

৭। একটি ধনাত্মক পূর্ণ সংখ্যার ফ্যাক্টরিয়াল মান নির্ণয়ের একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ব্যবহারিক অংশ দ্রষ্টব্য।

৮। শর্তসাপেক্ষে অসমবাহু ত্রিভুজক্ষেত্রের ক্ষেত্রফল নির্ণয়ের একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ব্যবহারিক অংশ দ্রষ্টব্য।

৯। ফাংশন ব্যবহার করে ৫০ পর্যন্ত মৌলিক সংখ্যাগুলো নির্ণয়ের প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ব্যবহারিক অংশ দ্রষ্টব্য।

১০। ফাংশন ওভারলোডিং প্রক্রিয়া ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ৪.৬ নং অনুচ্ছেদ অথবা, ব্যবহারিক অংশ দ্রষ্টব্য।।

১১। Constructor ও Destructor ফাংশন ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ৪.৭ নং অনুচ্ছেদ অথবা, ব্যবহারিক অংশ দ্রষ্টব্য।।

১২। প্রোগ্রামের মাধ্যমে Copy constructor এর ব্যবহার দেখাও।

উত্তর সংকেত ঃ ৪.৭ নং অনুচ্ছেদ অথবা, ব্যবহারিক অংশ দ্রষ্টব্য।।

১৩। মাল্টিলেভেল ইনহেরিটেন্স ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ৪.৭ নং অনুচ্ছেদ অথবা, ব্যবহারিক অংশ দ্রষ্টব্য।।

১৪। সিঙ্গেল ইনহেরিটেন্স ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ৪.৭ নং অনুচ্ছেদ অথবা, ব্যবহারিক অংশ দ্রষ্টব্য।।

১৫। হাইব্রিড ইনহেরিটেন্স ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ৪.৭ নং অনুচ্ছেদ অথবা, ব্যবহারিক অংশ দ্রষ্টব্য।

জাভা ও জাভার বৈশিষ্ট্য
Java & Features of Java

অধ্যায় ০৫

৫.০ ভূমিকা (Introduction) ঃ

অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর জগতে সর্বশ্রেষ্ঠ ও সর্বাধুনিক সংযোজন হচ্ছে জাভা (Java) ভাষা। জাভা একটি সত্যিকারের পূর্ণাঙ্গ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ। কারণ অবজেক্ট অরিয়েন্টেড ল্যাংগুয়েজের প্রধান তিনটি বৈশিষ্ট্য, যেমন– পলিমরফিজম (Polymorphism), ইনহেরিটেন্স (Inheritance) এবং এনক্যাপসুলেশন (Encapsulation) সহ অন্যান্য যেসব বৈশিষ্ট্য থাকা প্রয়োজন তার সবকিছুই জাভাতে আছে। এতে সি ও সি++ ভাষার অনেক জটিল ও বিঘ্ন সৃষ্টিকারী বিষয় যেমন- পয়েন্টার, অপারেটর ওভারলোডিং, মাল্টিপল ইনহেরিটেন্স ইত্যাদি বাদ দেয়ার পাশাপাশি সি ও সি++ ভাষায় সাপোর্ট করে না এমন অনেক নতুন নতুন বৈশিষ্ট্য যেমন- প্যাকেজ (Package), অ্যাপলেট (Applet) ও মাল্টিথ্রেড (Multithread) ইত্যাদি যোগ করা হয়েছে। ফলে জাভা সত্যিকার অর্থে একটি সহজ, সরল, সাবলীল, বিশ্বস্থ ও পূর্ণাঙ্গ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ভাষায় পরিণত হয়েছে।

৫.১ জাভা ভাষার ইতিহাস (History of Java) ঃ

কম্পিউটার আবিষ্কারের শুরু থেকে প্রোগ্রামিং এর বিভিন্ন পদ্ধতির প্রচলন ছিল। যেমন-

- মডুলার (Modular) প্রোগ্রামিং
- টপ ডাউন (Top-down) প্রোগ্রামিং
- বটম আপ (Bottom up) প্রোগ্রামিং ও
- স্ট্রাকচার্ড (Structured) প্রোগ্রামিং ইত্যাদি।

এসব পদ্ধতিগুলো তখনকার সময় ও চাহিদার প্রেক্ষিতে বেশ জনপ্রিয়ও ছিল। বিশেষত তুলনামূলক সহজ উপায়ে অপেক্ষাকৃত জটিল জটিল সমস্যার সমাধানে স্ট্রাকচার্ড প্রোগ্রামিং এর ব্যবহার ছিল খুবই গুরুত্বপূর্ণ। কিন্তু প্রযুক্তিগত উন্নয়ন ও চাহিদার কারণে এক সময়ে এসে সম্পূর্ণ ত্রুটিমুক্ত, সহজে রক্ষণাবেক্ষণযোগ্য ও পুনরায় ব্যবহারযোগ্য (Reusable) প্রোগ্রাম তৈরিতে সনাতন পদ্ধতিগুলো খুব বেশি কার্যকরী ভূমিকা পালন করতে পারছিল না। আর এসব সীমাবদ্ধতাতাকে অতিক্রম করার জন্যই নতুন ও যুগোপযোগী "অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং" এর আগমন। সিমুলা ৬৭ (Simula 67) হল প্রথম অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ যা ১৯৬০ সালে বাজারে আসে। এরপর আসে ভিএইচডিএল (VHDL)। পরবর্তীতে ১৯৮০ সালে যুক্তরাষ্ট্রের এটিএন্ডটি বেল ল্যাবরেটরি (AT&T, Bell laboratory) ও জর্ন স্ট্রাউসট্রাপ (Bjarne Stroustrup) এর যৌথ উদ্যোগে সি ভাষার প্রধান প্রধান বৈশিষ্ট্যের সাথে সিমলা ৬৭ ভাষার ক্লাস (Class) ধারণার সমন্বয়ে এক নতুন ভাষার জন্ম হয়– সি উইথ ক্লাস (C with class)। পরবর্তীতে একে মডিফাই করে ১৯৮৩ সালে নামকরণ করা হয় সি++ (C++), যা একটি জনপ্রিয় অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ। কিন্তু সি++ ভাষারও কিছু সীমাবদ্ধতা (Limitations) ছিল। যেমন– ইউজার ফ্রেন্ডলি না হওয়া, বিশ্বস্থতা (Reliability) ও পোর্টেবিলিটি (Portability) ইত্যাদি। এসব অসুবিধার জন্য প্রোগ্রামারগণ একটি পূর্ণাঙ্গ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ ডেভেলপ করার ক্ষেত্রে মনোযোগী হন।

ফলশ্রুতিতে ১৯৯১ সালে 'Sun Microsystems' কোম্পানির গবেষক James Gosling এর নেতৃত্বে একদল প্রোগ্রামার 'Green' প্রজেক্টের আওতায় সি++ ভাষার উপর ভিত্তি করে 'Consumer electronic device' এ ব্যবহার উপযোগী প্ল্যাটফর্ম ইনডিপেন্ডেন্ট (Platform independent) একটি ছোটখাট প্রোগ্রামিং ল্যাংগুয়েজ ডেভেলপ করার গবেষণা শুরু করেন এবং তারা একটি নতুন ভাষা উদ্ভাবন করেন যার নাম দেন 'Oak'। সম্ভবত Janes Gosling এর অফিস রুমের জানালার ধারে বিদ্যমান 'Oak' গাছটির কথা চিন্তা করেই এরূপ নামকরণ করে থাকবেন। কিন্তু 'Oak' নামের আরো একটি ল্যাংগুয়েজ বাজারে থাকায় এর নাম পরিবর্তন করে দেন জাভা (Java) এবং তা ১৯৯৪ সাল পর্যন্ত নানাভাবে মার্কেটিং এর চেষ্টা করেন। কিন্তু তাদের সে চেষ্টা ব্যর্থ হয়।

ইতিমধ্যে ইন্টারনেট ভিত্তিক ওয়ার্ল্ড ওয়াইড ওয়েব (World wide web-www) সিস্টেমের ব্যাপক প্রসার ঘটতে শুরু করে এবং এর জন্য 'mosaic' নামে একটি অ-বাণিজ্যিক ব্রাউজার (Browser) ব্যবহার করা হত। 'Sun microsystems' এর প্রোগ্রামারগণ তখন ল্যাংগুয়েজ হিসেবে জাভার ক্ষমতা প্রদর্শনের লক্ষ্যে জাভা ভাষা ব্যবহার করেই জাভা ভার্চুয়াল মেশিন বা 'JVM' ব্যবস্থা সমৃদ্ধ 'Hot Java' নামে একটি ব্রাউজার নির্মাণ করলেন। ১৯৯৫ সালের ২৩ মে তারিখে এটি 'Sunworld' নামক অনলাইন ম্যাগাজিনে প্রথম রিলিজ করেন।

ব্রাউজার হিসেবে জাভা সফল হওয়ায় ১৯৯৫ সালেই Netscape এবং পরবর্তীতে IBM, Symantec, Inprise এমনকি Microsoft সহ বহু প্রতিষ্ঠান তাদের ব্রাউজারে জাভা ব্যবহার করা শুরু করেন। তাই জাভাকে সি++ এর ইন্টারনেট ভার্সনও বলে।

১৯৯৬ সালে জাভার প্রথম ভার্সন প্রকাশিত হয়। প্রথম ভার্সন প্রকাশের কয়েক মাস পরই পরবর্তী ভার্সন (1.02) প্রকাশিত হয়। পরবর্তীতে বড় বড় মাপের প্রোগ্রাম লেখার সুবিধা নিয়ে লাইব্রেরি ফাংশনসমৃদ্ধ ভার্সন (1.1) প্রকাশিত হয়। সময়ের ধারাবাহিকতায় 1.1 ভার্সনের তুলনায় উন্নততর এবং অনেক বেশি বিল্ট ইন ক্লাস সমৃদ্ধ ভার্সন 1.2 প্রকাশিত হয় ১৯৯৮ সালে। ভার্সনটি রিলিজ হওয়াব মাত্র তিন দিন পরেই এর নামকরণ করা হয় Java 2। Java 2 রিলিজ হওয়ার পরবর্তী সময়ে জাভার আরো কয়েকটি ভার্সন (JDK 1.3, JDK 1.4, JDK 1.5) ইত্যাদি পর্যায়ক্রমে প্রকাশিত হয়।

৫.১.১ জাভার বৈশিষ্ট্য (Features of Java) ঃ

জাভা ডেভেলপারগণ চেয়েছিলেন প্রচলিত প্রসিডিউর ও অবজেক্ট অরিয়েন্টেড প্রোগ্রাম পদ্ধতির সমস্যাগুলো কাটিয়ে জাভাকে একটি সহজ, সাবলীল, বিশ্বস্ত, পোর্টেবল ও গ্রহণযোগ্য অবজেক্ট অরিয়েন্টেড প্রোগ্রাম ভাষায় পরিণত করতে। Java'র নির্মাতাগণ তাদের এই language এর design এবং লক্ষ্য ও উদ্দেশ্য ব্যাখ্যা করার জন্য ১০ টি buzzword সমৃদ্ধ একটি শ্বেতপত্র প্রকাশ করেন। এগুলোই জাভার বৈশিষ্ট্য। বৈশিষ্ট্যগুলো হচ্ছে–

- অবজেক্ট অরিয়েন্টেড (Object Oriented)
- কম্পাইল্ড ও ইন্টারপ্রেটেড (Both Compiled and Interpreted)
- প্লাটফরম অনির্ভরশীলতা (Platform Independent)
- পোর্টেবল (Portable)
- উচ্চ ক্ষমতাসম্পন্ন (Highly Efficient)
- সহজ, সাবলীল ও ছোট (Easier, Simple and Small)
- শক্তিশালী ও নিরাপদ (Robust and Secure)
- গতিশীল ও সম্প্রসারণযোগ্য (Dynamic and Extensible)
- ডিস্ট্রিবিউটেড (Distributed)
- মাল্টিথ্রেডেড (Multithreaded)

অবজেক্ট অরিয়েন্টেড (Object Oriented) ঃ জাভা একটি সত্যিকার অবজেক্ট অরিয়েন্টেড প্রোগ্রাম ভাষা। সুতরাং এর সবকিছুই ক্লাস, মেথড ও অবজেক্ট নিয়ন্ত্রিত। অর্থাৎ প্রোগ্রামের নির্বাহযোগ্য সকল স্টেটমেন্ট ক্লাসের মধ্যে লেখা হয়। ইনহেরিটেন্স প্রক্রিয়ায় প্রয়োজনে পুরাতন এক বা একাধিক ক্লাসের বৈশিষ্ট্যর আলোকে নতুন নতুন ক্লাস তৈরি করা যায়। এ প্রক্রিয়ায় এক ক্লাস বা অবজেক্টের বৈশিষ্ট্য অপর ক্লাস বা অবজেক্টে স্থানান্তরিত হয়। জাভা প্যাকেজে বিপুল সংখ্যক ক্লাসের সমাবেশ ঘটান যায়। Object Oriented design মূলত প্রোগ্রাম রচনার একটি টেকনিক। যে ল্যাংগুয়েজের মাধ্যমে এই পদ্ধতিতে প্রোগ্রাম লেখা হবে সেই ল্যাঙ্গুয়েজ কে অবশ্যই OOP সাপোর্ট করতে হবে। Java সম্পূর্ণরূপে OOP সাপোর্ট করে। Polymorphism, Inheritance এবং Encapsulation সহ একটি OOP language এর যা কিছু গুণাগুণ থাকা প্রয়োজন তার সবকিছুই জাভাতে আছে। সুতরাং জাভা কে নির্দ্বিধায় বলা যায় এটি Object Oriented language।

কম্পাইল্ড ও ইন্টারপ্রেটেড (Both Compiled and Interpreted) ঃ প্রোগ্রাম ভাষাগুলো সাধারণত কম্পাইলার অথবা ইন্টারপ্রিটার অনুবাদিত হয়। কম্পাইলার অনুবাদিত প্রোগ্রাম ভাষাগুলো মেশিন নির্ভর হয়, অর্থাৎ এক ধরনের মেশিন ও অপারেটিং সিস্টেম কর্তৃক অনুবাদিত প্রোগ্রাম ভিন্ন ধরনের মেশিন ও অপারেটিং সিস্টেমে রান করে না। কিন্তু জাভা কম্পাইলার ও ইন্টারপ্রেটার উভয় কর্তৃক অনুবাদযোগ্য। এজন্য জাভাকে একটি দ্বি-পর্যায় (Two-stage) বিশিষ্ট প্রোগ্রাম ভাষা বলা হয়। প্রথম পর্যায়ে সোর্স প্রোগ্রাম কম্পাইলার কর্তৃক অনুবাদিত হয়ে বাইটকোডে (bytecode) রূপান্তরিত হয়। দ্বিতীয় পর্যায়ে জাভা ইন্টারপ্রিটার কর্তৃক জাভা প্রোগ্রাম ব্যবহারকারী মেশিনের বোধগম্য মেশিনকোডে পরিণত হয়। এজন্য জাভাকে কম্পাইলার এবং ইন্টারপ্রিটার উভয় কর্তৃক অনুবাদিত একটি যন্ত্র নিরপেক্ষ প্রোগ্রাম ভাষা বলা হয়।

প্লাটফরম অনির্ভরশীলতা (Platform Independent) ঃ জাভা ভাষার সবচেয়ে আকর্ষণীয় এবং অন্যতম বৈশিষ্ট্য হল এর প্লাটফরম অনির্ভরশীলতা। জাভা কোন নির্দিষ্ট হার্ডওয়্যার, সফ্টওয়্যার ও অপারেটিং সিস্টেম নির্ভরশীল প্রোগ্রাম ভাষা নয়। অর্থাৎ এক ধরনের মেশিনে কম্পাইল এবং নির্বাহ করা জাভা প্রোগ্রাম সহজেই অন্য ধরনের মেশিনে নির্বাহ করা যায়। মূলত জাভা কম্পাইলার যে বাইটকোড তৈরি করে তা যে কোন হার্ডওয়্যার, সফ্টওয়্যার ও অপারেটিং সিস্টেম বিশিষ্ট কম্পিউটারে নির্বাহযোগ্য। আবার প্রিমিটিভ ডাটা টাইপগুলোর সাইজও মেশিন নির্ভর নয়। অর্থাৎ সি++ এ যেমন int টাইপের জন্য কখনও ২ বাইট আবার কখনও ৪ বাইট (মডিফায়ার ব্যবহার করলে) মেমোরি বরাদ্দ হয়, কিন্তু জাভায় int টাইপের জন্য সর্বদাই ৪ বাইট মেমোরি বরাদ্দ হয়। উভয় কারণে জাভা কোন নির্দিষ্ট প্লাটফরম নির্ভরশীল প্রোগ্রাম ভাষা নয়।

পোর্টেবল (Portable) ঃ জাভা ভাষার অপর একটি আকর্ষণীয় বৈশিষ্ট্য হল এর পোর্টেবিলিটি। জাভা কোন নির্দিষ্ট প্লাটফরম নির্ভর প্রোগ্রাম ভাষা না হওয়ায় এক মেশিনে কম্পাইল এবং নির্বাহ করা জাভা প্রোগ্রাম সহজেই অন্য মেশিনে নির্বাহ করা যায়। এজন্য ব্রাউজিং প্রক্রিয়ায় লোকাল (স্থানীয়) কম্পিউটার থেকে রিমোট (দূরবর্তী) কম্পিউটারে রক্ষিত জাভা প্রোগ্রাম নির্বাহ করা যায় কিংবা প্রয়োজনে ডাউনলোড করে ব্যবহার করা যায়। মূলত এরূপ বৈশিষ্ট্যই জাভাকে সার্বজনীন গ্রহণযোগ্য নেটওয়ার্ক প্রোগ্রাম ভাষায় পরিণত করেছে।

উচ্চ ক্ষমতাসম্পন্ন (Highly Efficient) ঃ যদিও জাভা প্রোগ্রাম অনুবাদে ইন্টারপ্রিটারের ব্যবহার অনেকটা সেকেলে মনে হয়। কিন্তু প্রাথমিক পর্যায়ে কম্পাইলার কর্তৃক জাভা সোর্স প্রোগ্রাম অনুদিত হয়ে যে বাইটকোডে (bytecode) রূপান্তরিত হয় তা নির্দিষ্ট কোন মেশিন নির্ভর না হওয়ায় পরবর্তী পর্যায়ে ইন্টারপ্রিটারের ব্যবহার জাভা রান-টাইম সিস্টেমের গতি বাড়িয়ে দিয়েছে বহুগুণে। আর রান-টাইম সিস্টেমের অতিরিক্ত সময় হ্রাসের বিষয়টি লক্ষ্য রেখেই জাভা আর্কিটেকচার গঠিত হয়েছে। সান কর্পোরেশনের মতে, সি ও সি++ এর তুলনায় জাভা অতি উচ্চ ক্ষমতাসম্পন্ন। তাছাড়া প্যাকেজ, মাল্টিথ্রেড প্রভৃতি বিষয়ের সমাহার জাভা ভাষার ক্ষমতা বাড়িয়ে দিয়েছে বহুগুণে।

সহজ, সাবলীল ও ছোট (Easier, Simple and Small) ঃ জাভা সহজ, সরল, সাবলীল ও ছোট্ট একটি প্রোগ্রাম ভাষা। জাভা প্রোগ্রাম কৌশল অনেকটা সি এবং সি++ এর মত। উপরন্তু সি ও সি++ এর অনেক জটিল বিষয় (যেমন– পয়েন্টার, অপারেটর ওভারলোডিং, মাল্টিপল ইনহেরিটেন্স, ইত্যাদি) জাভায় বাদ দেওয়া কিংবা অনেক ক্ষেত্রে এগুলোর বিকল্প ব্যবস্থা রাখা হয়েছে। আবার জাভায় সি ও সি++ অনুমোদন করে না এরূপ অনেক বৈশিষ্ট্য (যেমন– লেবেলযুক্ত continue ও goto স্টেটমেন্ট, প্যাকেজ, মাল্টিথ্রেড ও অ্যাপলেট ইত্যাদি) সংযোজন করা হয়েছে। উভয় কারণে জাভা সর্বজন স্বীকৃত একটি সহজ, সরল, সাবলীল ও ছোট্ট প্রোগ্রাম ভাষা হিসেবে পরিচিতি পেয়েছে। তবে জাভা কম্পাইলার তুলনামূলকভাবে ছোট হলেও এর পরিধি এবং কার্যক্ষমতা অতি বিশাল ও ব্যাপক। জাভা প্রোগ্রাম কোড দেখতেও অনেকটা সি++ এর মত, এজন্য জাভাকে সি++ এর সরল সংস্করণ বলা হয়।

শক্তিশালী ও নিরাপদ (Robust and Secure) ঃ জাভা একটি শক্তিশালী ও নিরাপদ প্রোগ্রাম ভাষা। জাভায় সংযোজিত হয়েছে এমন কতগুলো বৈশিষ্ট্য যেগুলো বিভিন্নভাবে প্রোগ্রাম কোড ও রান-টাইম সিস্টেমকে নিরাপদ রাখতে সক্ষম। যেমন– জাভা কম্পাইল টাইম এবং রান-টাইম উভয় পর্যায়ে ডাটা টাইপ পরীক্ষা করে, এর গারবেজ কালেক্টর স্বয়ংক্রিয়ভাবে সিস্টেমের মেমোরি ব্যবস্থাপনা সম্পর্কিত যাবতীয় সমস্যার মোকাবিলা করে, জাভা পুরোপুরিভাবে এক্সেপশন হ্যান্ডলিং (exception handling) প্রক্রিয়ার সাথে একত্রীভূত যা ধারাবাহিকভাবে প্রোগ্রামের এরর হস্ত ক্ষেপপূর্বক প্রোগ্রাম ও সিস্টেমকে নিরাপদ রাখে ইত্যাদি। নেটওয়ার্ক ও ইন্টারনেটে ব্যবহারযোগ্য প্রোগ্রাম ভাষা হওয়ায় জাভা অনাকাঙ্ক্ষিত ও অননুমোদিত ব্যবহারকারী ও কম্পিউটার ভাইরাস থেকে নিরাপদ থাকা আবশ্যক। মূলত একারণেই জাভায় পয়েন্টার ভেরিয়েবল ব্যবহার উহ্য রাখা হয়েছে। ফলে জাভা অবাঞ্ছিত মেমোরি এক্সেস ও ভাইরাস থেকে নিরাপদ থাকে। জাভার শুরু থেকেই Security বা নিরাপত্তা দিকটিতে এর নির্মাতাদের ছিল বিশেষ দৃষ্টি। এ ছাড়াও প্রতিনিয়ত এর Security bug গুলো খুঁজে বের করা হচ্ছে এবং bug গুলো সংশোধন করা হচ্ছে। এসব কারণে Java অন্য যে কোন Programming language এর তুলনায় অনেক বেশি Secure । Java তে রচিত Program সবসময়ই Virus-free এবং tamper-free ।

গতিশীল ও সম্প্রসারণযোগ্য (Dynamic and Extensible) ঃ জাভা একটি গতিশীল ভাষা যা স্বয়ংক্রিয়ভাবে নতুন নতুন ক্লাস, অবজেক্ট, মেথডের সাথে সংযুক্ত হতে সক্ষম। জাভা ইন্টারপ্রিটার প্রোগ্রাম নির্বাহকালীন সময়ে কোয়েরি (query) সম্পাদনের মাধ্যমে কাঙ্ক্ষিত ক্লাস, অবজেক্ট, মেথড, থ্রেড ইত্যাদির সাথে সংযোগ স্থাপন কিংবা প্রয়োজনে সংযোগ বাতিল এমনকি প্রোগ্রাম নির্বাহ স্থগিত কিংবা বাতিল করতে সক্ষম। উপরন্তু, জাভা রান-টাইম সিস্টেম স্বয়ংক্রিয়ভাবে নেটিভ ফাংশন (native function) এর সাথে সংযুক্ত হতে পারে যা জাভাকে একটি সম্প্রসারিত প্রোগ্রাম ভাষায় রূপদান করেছে। এ প্রক্রিয়ায় জাভা ভাষায় সি, সি++ এর মত অন্যান্য প্রোগ্রাম ভাষায় লেখা কার্যকরী ফাংশন ব্যবহার করা যায়। Interpreted language হিসেবে Java আসলেই একটি High performance language। যেহেতু জাভা একটি মধ্যবর্তী বাইট কোড কে ইন্টারপ্রিট করে কাজেই এর গতি অন্য যে কোন ইন্টারপ্রিটেড প্রোগ্রামের তুলনায় বেশি। তবে সি++ বা ভিজুয়্যাল বেসিক, মেশিন নির্ভর কম্পাইলড কোডে চলে বিধায় জাভার তুলনায় বেশি গতিসম্পন্ন।

ডিস্ট্রিবিউটেড (Distributed) ঃ মূলত নেটওয়ার্ক সিস্টেমে ব্যবহারের উদ্দেশ্যে একটি ডিস্ট্রিবিউটেড প্রোগ্রাম ভাষা হিসেবে জাভার প্রসার ঘটে। জাভা প্রোগ্রাম একই সাথে স্থানীয় (লোকাল) এবং দূরবর্তী (রিমোট) কম্পিউটার থেকে ডাটা ও কোড এক্সেস করতে সক্ষম। ফলশ্রুতিতে বিভিন্ন স্থানে অবস্থিত একাধিক প্রোগ্রামার একটি সিঙ্গেল প্রজেক্ট প্রোগ্রামে সম্মিলিতভাবে কাজ করতে সক্ষম হন। জাভার নেটওয়ার্কিং ক্ষমতা শক্তিশালী এবং সহজ। Socket connection open করার মতো নেটওয়ার্কিং-এর অনেক কাজ জাভাতে সহজে করা সম্ভব যা অন্য কোন language এ সম্ভব নয়। Servlets এর মাধ্যমে Server side processing অনেক বেশি দক্ষতার সাথে করা সম্ভব। ইদানিং অনেক জনপ্রিয় Web server 'Servlets' সাপোর্ট করে। এ ছাড়াও রিমোট মেথড ইনভোকেশন বা RMI পদ্ধতিতে distributed object এর মধ্যে যোগাযোগ সম্ভব।

মাল্টিথ্রেডেড (Multithreaded) ঃ জাভা একটি মাল্টিথ্রেডেড প্রোগ্রাম ভাষা যা একই সাথে একাধিক কার্য সম্পাদনে সক্ষম। এজন্য কোন এপ্লিকেশন প্রোগ্রামে নতুন কোন কার্য সম্পাদনের জন্য পূর্বে নির্দেশ দেয়া হয়েছে এমন কার্য সম্পাদনের জন্য অপেক্ষা করতে হয় না। জাভা মাল্টিথ্রেডিং সিস্টেমে একই সাথে একাধিক কার্য সম্পাদন করা যায়। যেমন– লোকাল সিস্টেমে গান শোনার পাশাপাশি একই সময়ে দূরবর্তী কম্পিউটার থেকে কিছু ডাউনলোড করা যায় কিংবা পূর্বে ডাউনলোড করা ফাইল সার্চ বা ভিউ করা যায়। এ প্রক্রিয়া অনেকটা মাল্টিটাস্কিং (multitasking) সিস্টেমের মত। উইন্ডোজ এবং উইন্ডোজ ভিত্তিক অনেক অ্যাপ্লিকেশন মাল্টিটাস্কিং সিস্টেমের অন্তর্গত। মাল্টিথ্রেড প্রক্রিয়ায় মূলত পারস্পরিক সক্রিয় দুই বা ততোধিক গ্রাফিক্যাল অ্যাপ্লিকেশন সিস্টেমের যুগপৎ নির্বাহ ক্ষমতা বহুগুণ বৃদ্ধি করেছে।

৫.২ জাভা ডেভেলপমেন্ট এনভায়রনমেন্ট এর ধাপসমূহ (Java Development Environment Steps) ঃ

জাভা ভাষায় প্রোগ্রাম রচনা (Develop) ও কম্পাইল (Compile) করার জন্য নিম্নবর্ণিত ধাপসমূহ অনুসরণ করতে হয়। যেমন-

- জাভা কম্পাইলার ইন্সটলেশন (JDk Installation)
- সঠিক পাথ সেটিং (Path setting)
- প্রোগ্রাম কোডিং (Program coding)
- প্রোগ্রাম কম্পাইলিং (Program compiling)
- প্রোগ্রাম রান করা (Program running) ইত্যাদি।

জাভা কম্পাইলার ইনস্টলেশন (JDK Installation) ঃ জাভা ভাষায় ব্যবহৃত কম্পাইলারকে জাভা ডেভেলপমেন্ট কিট (Java development kit) বা সংক্ষেপে JDK বলে। প্রোগ্রাম রচনার জন্য প্রয়োজনীয় টুল (Tool) এবং সহস্রাধিক ক্লাস ও মেথডের (ফাংশন) সমন্বয়ে গঠিত কম্পাইলারই হচ্ছে JDK (Java development kit)। এর বেশকিছু ভার্সন আছে। যেমন- JDK 1.01, JDk 1.02, JDK 1.1, JDK 1.2, JDK 1.3, JDK 1.4, JDK 1.5 ইত্যাদি।

JDK পুরো সিস্টেমটাই কমপ্রেসড আকারে থাকে। তাই কমপ্রেসকৃত ঐ ফাইলে (JDk 1.3) ডাবল ক্লিক করলেই জাভা ইন্সটল শুরু হবে। তবে ইন্সটল করার সময় ড্রাইভ (C :, D :, E : ইত্যাদি) নির্দিষ্ট করে দেয়া যায়। সাধারণত C : ড্রাইভকে ডিফল্ট ড্রাইভ হিসেবে নির্দিষ্ট করা থাকে। তাই অন্য কোন ড্রাইভ সিলেক্ট না করলে কম্পাইলারটি C ড্রাইভেই ইন্সটল হবে।

জাভা প্রোগ্রাম কম্পাইল, ডিবাগ ও রান করার জন্য JDK (Java development kit) এর মধ্যে বেশকিছু সাব ডিরেক্টরি ও টুল বিদ্যমান।

নিম্নে \jdk ডিরেক্টরির অধীনে বিদ্যমান সাব ডিরেক্টরির তালিকা উল্লেখ করা হল।

সাব ডিরেক্টরি	বিবরণ
docs	এই সাবডিরেক্টরিতে HTML ফরম্যাটে লাইব্রেরি ডকুমেন্টসগুলো বিদ্যমান।
bin	bin সাবডিরেক্টরিতে কম্পাইলার এবং কম্পাইলিং টুলগুলো বিদ্যমান।
demo	এই সাবডিরেক্টরিতে কিছু demo ফাইল আছে।
Include	Native method এর ফাইলসমূহ Include সাবডিরেক্টরিতে বিদ্যমান।
lib	এই সাবডিরেক্টরিতে লাইব্রেরি ফাইলসমূহ বিদ্যমান।
src	লাইব্রেরি ফাংশনসমূহের সোর্স ফাইলগুলো src সাবডিরেক্টরিতে বিদ্যমান।

টেবিল ঃ জাভার সাব ডিরেক্টরি তালিকা

JDk তে বিদ্যমান টুলসমূহ ও তাদের ব্যবহার নিম্নরূপ ঃ

টুলস	ব্যবহার
appletviewer	applet রান করার জন্য এটি ব্যবহৃত হয়।
java	জাভা ইন্টারপ্রিটার (Java Interpreter), এটি সোর্স Bytecode কে মেশিনের জন্য Interpret করে।
javac	জাভা কম্পাইলার (Java compiler), এটি সোর্স কোডকে Byte code এ রূপান্তরিত করে (compile করে)
javadoc	Source থেকে HTML format এ documentation তৈরি করে।
javah	Native মেথডে ব্যবহারের জন্য header file তৈরি করে।
javap	Byte code- কে আবার source কোডে ফিরিয়ে আনে।
jdb	Java debugger

টেবিল ঃ জাভা টুলস

সঠিক পাথ সেটিং (Path setting) ঃ সঠিক পাথ (path) সেটিং এর জন্য Dos prompt-এ গিয়ে টাইপ করতে হবে-

edit autoexec. bat

তারপর এন্টার কী চাপলে autoexec ফাইলটি ডিসপ্লে হবে।

অতঃপর নিম্নরূপ টাইপ করতে হবে-

path = c :\jdk 1.3\bin;

সবশেষে File → Save কমান্ডের মাধ্যমে ফাইলটি সেভ করে কম্পিউটারটি রিস্টার্ট করলে উক্ত কম্পিউটারটি জাভা প্রোগ্রাম রচনা ও কম্পাইল করার জন্য তৈরি হবে।

প্রোগ্রাম কোডিং (Program coding) ঃ জাভা সোর্স কোড লেখার জন্য পছন্দমত একটি টেক্সট এডিটর (Text editor) যেমন- U ltra edit 6.0, Text pad ইত্যাদি ইন্সটল করতে হবে। অথবা Windows Notepad ও সোর্সকোড (source code) লেখার জন্য ব্যবহার করা যেতে পারে। সোর্স কোড লেখার পর Java এক্সটেনশন (যেমন- start. Java, text. Java) ব্যবহার করে কোন নামে ফাইলটি সেভ করতে হবে। একটি জাভা সোর্স ফাইলে যদি একাধিক ক্লাস থাকে তবে যে ক্লাসটিতে main ( ) মেথড থাকবে সেই ক্লাসের নামেই হবে জাভা সোর্স ফাইলের নাম।

প্রোগ্রাম কম্পাইলিং (Program compiling) ঃ সোর্সকোডকে মেশিনকোড বা বাইটকোডে রূপান্তর করার প্রক্রিয়াকে প্রোগ্রাম কম্পাইলিং বলে। প্রোগ্রাম কম্পাইল করার জন্য 'Javac' কমান্ডটি ব্যবহার করা হয়। তবে এজন্য Dos prompt-এ গিয়ে সোর্স ফাইলটি যে লোকেশনে সেভ করা হয়েছে। যেমন- C : \jdk 1.3\bin অথবা ক্ষেত্র বিশেষে অন্য কোন লোকেশন) সেই লোকেশনে যেতে হবে। তারপর নিম্নরূপ কমান্ড প্রয়োগ করতে হবে।

C : \jdk 1.3\bin > javac start. java [ENTER]

এখানে 'start' হচ্ছে সোর্স ফাইলের নাম।

প্রোগ্রাম রান করা (Program running) ঃ জাভা প্রোগ্রাম রান করার জন্য Java.exe ইন্টারপ্রিটারটি ব্যবহৃত হয়। এটি JDK এর 'bin' সাব ডিরেক্টরিতে বিদ্যমান। জাভা প্রোগ্রাম রান করার ক্ষেত্রে প্রোগ্রামটি কম্পাইল করার পর রান করতে হয়।

C : \jdk 1.3\bin > javac start. java [ENTER]// কম্পাইলিং

C : \jdk 1.3\bin> java start [ENTER] // রানিং

উভয়ক্ষেত্রেই 'start' হচ্ছে সোর্স ফাইলের নাম।

৫.২.১ জাভা ভার্চুয়াল মেশিন (Virtual machine of Java-JVM) ঃ

যে কোন হাই লেভেল ল্যাংগুয়েজ-এর সোর্স কোডগুলোকে মেশিনের বোধগম্য ভাষা বা মেশিন কোডে রূপান্তরিত করে রান বা এক্সিকিউট করতে হয়। যেহেতু এ মেশিন কোড একেক মেশিনের জন্য একেক রকম, কাজেই প্রতিটি আলাদা ধরনের মেশিনের জন্য সোর্স কোডগুলোকে আলাদাভাবে কম্পাইল করতে হয়।

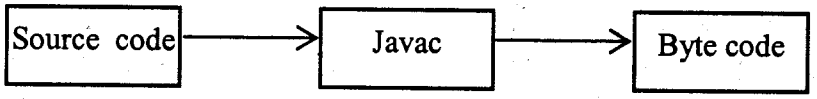
জাভা কম্পাইলার ও এ কাজটিই করে অর্থাৎ সোর্স কোডকে মেশিন কোডে রূপান্তরিত করে। জাভা কম্পাইলার সোর্স কোডকে যে মেশিনের জন্য কম্পাইল করে সেটি আসলে বাস্তবে মেশিন নয়, এটি একটি virtual machine, এটি অবস্থান করে কম্পিউটার এর মেমোরিতে। এটি যেন কম্পিউটার মেশিনের ভেতর আর একটি কম্পিউটার মেশিন। একটি বাস্তব কম্পিউটারের প্রায় সমস্ত গুরুত্বপূর্ণ কাজ এ ভার্চুয়াল মেশিন সমাধা করে। এটিকে বলা হয় জাভা ভার্চুয়াল মেশিন (Java Vritual Machine) সংক্ষেপে JVM।

সুতরাং জাভা ভার্চুয়াল মেশিন (JVM) হচ্ছে এমন একটি ভার্চুয়াল মেশিন যা জাভা ভাষায় লিখিত প্রোগ্রাম বা সোর্স কোডকে প্রথমে বাইট কোডে (Byte code) ও পরবর্তীতে মেশিন কোডে (Machine code)-এ রূপান্তর করে।

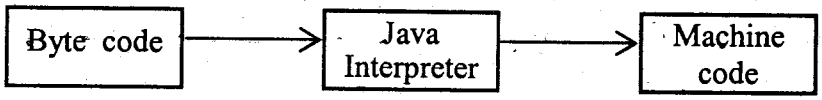
কম্পিউটার যে প্ল্যাটফর্মেরই হোক না কেন যদি এতে JVM লোড করা থাকে তবে সে মেশিনে জাভা প্রোগ্রাম চলবে।

জাভা ভার্চুয়াল মেশিন বা JVM দুটো ধাপে (stage) এর কার্য সম্পাদন করে থাকে।

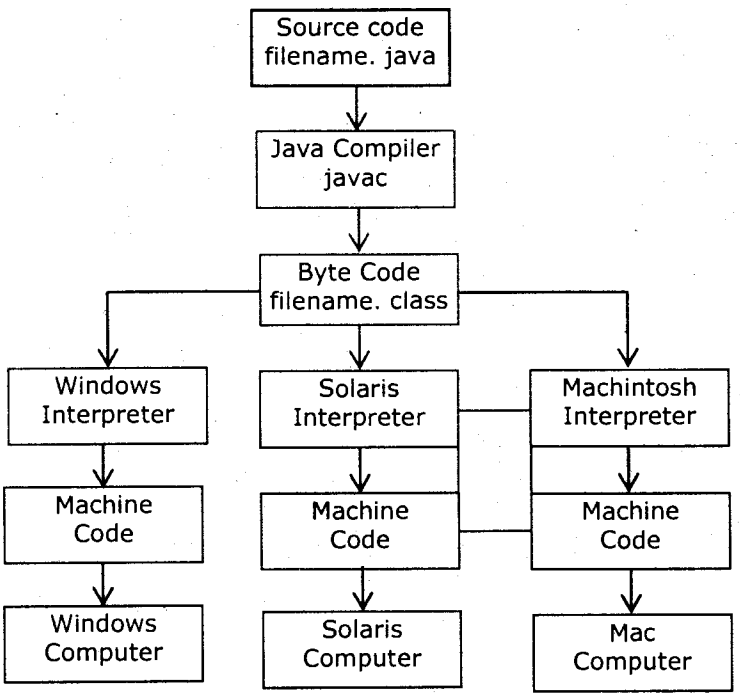
১ম ধাপ ঃ সোর্স কোডকে অর্থাৎ জাভা ভাষায় লিখিত প্রোগ্রামকে কম্পাইলার ব্যবহার করে ভার্চুয়াল মেশিনের জন্য বাইট কোডে (Byte code) রূপান্তর করে।



২য় ধাপ ঃ দ্বিতীয় ধাপে ভার্চুয়াল মেশিন বাইট কোডকে জাভা ইন্টারপ্রিন্টার ব্যবহার করে মেশিন উপযোগী মেশিন কোডে (Machine code) রূপান্তর করে।



বাইট কোড মেশিন স্পেসিফিক (Machine specific) নয়। তাই JVM ভার্চুয়াল মেশিন ও রিয়েল মেশিনের মাঝে অবস্থান করে মেশিন স্পেসিফিক মেশিন কোড জেনারেট করে। সেজন্য JVM একেক ধরনের মেশিনের জন্য একেক ধরনের ইন্টারপ্রিন্টার যেমন- উইন্ডোজ মেশিনের জন্য উইন্ডোজ ইন্টারপ্রিন্টার, ম্যাকিন্টোশ মেশিনের জন্য ম্যাকিন্টোশ ইন্টারপ্রিন্টার, সোলারিজ (Solaries) মেশিনের জন্য সোলারিজ ইন্টারপ্রিন্টার ব্যবহৃত হয় যা নিম্নের চিত্রের মাধ্যমে দেখানো হল।



চিত্র ঃ

৫.৩ জাভা ও সি++ ল্যাংগুয়েজের মধ্যে পার্থক্য (Differences between Java and C++) ঃ

জাভা ভাষায় সি++ এর বিঘ্ন সৃষ্টিকারী বৈশিষ্ট্যাবলি বাদ দিয়ে অবজেক্ট অরিয়েন্টেড প্রোগ্রাম পদ্ধতির নতুন অনেক বৈশিষ্ট্যের সমন্বয় ঘটান হয়েছে, ফলে সত্যিকার অর্থে জাভা একটি সহজ, সরল, সাবলীল, বিশ্বস্ত ও পোর্টেবল এবং পূর্ণাঙ্গ অবজেক্ট অরিয়েন্টেড প্রোগ্রাম ভাষায় পরিণত হয়েছে। এজন্য জাভাকে সি++ এর সাব-সেটও বলা হয় না আবার সুপার-সেটও বলা হয় না। তবে জাভায় সি++ অপেক্ষা ইন্টারনেট প্রোগ্রামিং কৌশল প্রাধান্য পাওয়ায় একে সি++ এর ইন্টারনেট ভার্সন বলা হয়।

নিম্নে জাভা ও সি++ ল্যাংগুয়েজ পার্থক্য উল্লেখ করা হল ঃ

জাভা	সি++
জাভা একটি পূর্ণাঙ্গ অবজেক্ট অরিয়েন্টেড ল্যাংগুয়েজ যার মাধ্যমে তিন ধরনের প্রোগ্রাম (কনসোল মোড অ্যাপ্লিকেশন (Console mode application), জি ইউ আই অ্যাপ্লিকেশন (GUI Application) ও অ্যাপেলেট ইত্যাদি)। রচনা করা যায়।	সি++ ও অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ যার মাধ্যমে দুই ধরনের প্রোগ্রাম (কনসোল মোড অ্যাপ্লিকেশন (Consol mode application ও জিইউআই অ্যাপ্লিকেশন) রচনা করা যায়।
জাভা প্রোগ্রামে কোন হেডার ফাইল প্রয়োজন হয় না।	সি++ প্রোগ্রামে হেডার ফাইল অত্যাবশ্যকীয়।
এটি প্রিপ্রসেসর ডিরেক্টিভ (যেমন- # include, # define) অনুমোদন করে না।	সি++ ভাষায় প্রিপ্রসেসর ডিরেক্টিভ ম্যাক্রোর ব্যবহার অনুমোদন করে।

জাভা	সি++
অন্য ভাষায় লেখা ফাংশন জাভা প্রোগ্রাম সাপোর্ট করে।	অন্য ভাষায় লেখা ফাংশন সি++ প্রোগ্রাম সাপোর্ট করে না।
জাভায় পয়েন্টার, স্ট্রাকচার ও ইউনিয়ন ব্যবহারের সুযোগ নেই।	সি++ এ পয়েন্টার, স্ট্রাকচার ও ইউনিয়ন ব্যবহারের সুযোগ আছে।
জাভা ফাংশন ওভারলোডিং সাপোর্ট করে কিন্তু অপারেটর ওভারলোডিং সাপোর্ট করে না।	সি++ ফাংশন ওভালোডিং ও অপারেটর ওভারলোডিং উভয়ই সাপোর্ট করে।
জাভা যেহেতু সোর্স কোডকে প্রথমে বাইট কোড এবং পরে বাইট কোডকে মেশিন কোডে রূপান্তরিত করে কাজ করে তাই সি++ এর তুলনায় এটি কম গতি সম্পন্ন।	সি++ যেহেতু সরাসরি সোর্স কোডকে মেশিন কোডে রূপান্তর করে তাই জাভার তুলনায় এটি অধিক গতিসম্পন্ন।
জাভা গ্লোবাল ভেরিয়েবলের ব্যবহার অনুমোদন করে না। এতে ব্যবহৃত সকল ভেরিয়েবল কোন না কোন ক্লাসের মেম্বার (লোকাল) ভেরিয়েবল হিসেবে ঘোষণা করা হয়।	সি++ গ্লোবাল ভেরিয়েবলের ব্যবহার অনুমোদন করে।
জাভায় ইন্টিজার টাইপ ডাটা সর্বদাই ৩২ বিটের হয়। অর্থাৎ জাভা signed ও unsigned টাইপ মডিফায়ার ব্যবহার অনুমোদন করে না।	সি++ এ ইন্টিজার টাইপ ডাটা ১৬ বিট বা ৩২ বিট উভয় ধরনেরই হয়। অর্থাৎ signed ও unsigned টাইপ মডিফায়ার ব্যবহার অনুমোদন করে।
জাভায় মাল্টিপল ইনহেরিটেন্স এর বিকল্প হিসেবে ইন্টারফেস প্রক্রিয়া ব্যবহৃত হয়।	সি++ মাল্টিপল ইনহেরিটেন্স প্রক্রিয়া অনুমোদন করে।
জাভায় ডেস্ট্রাক্টর মেথডের পরিবর্তে ফাইনালাইজার মেথড ব্যবহৃত হয়।	সি++ এ ডেস্ট্রাক্টর মেথড ব্যবহৃত হয়।
জাভা ডিফল্ট অ্যারগুমেন্টেড মেথডের ব্যবহার অনুমোদন করে না।	সি++ আরগুমেন্টেড মেথডের ব্যবহার অনুমোদন করে।

৫.৪ জাভার প্রয়োগক্ষেত্র বা ব্যবহার (Applications of Java) ঃ

জাভা একটি পরিপূর্ণ অবজেক্ট-অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ হিসেবে এর মাধ্যমে সাধারণ কাজে ব্যবহৃত প্রায় সকল ধরনের প্রোগ্রাম রচনার পাশাপাশি বাস্তবভিত্তিক বিভিন্ন প্রকারের জটিল সমস্যারও সমাধান করা যায়। প্রোগ্রামিং এর নানা ক্ষেত্রে জাভা ভাষার ব্যবহার ক্রমান্বয়ে বেড়েই চলছে। বিশেষ করে ইউজার ইন্টানফেস ডিজাইন, উইন্ডোজ ভিত্তিক প্রোগ্রাম, রিয়েল টাইম সিস্টেম, সিমুলেশন ও মডেলিং, অবজেক্ট অরিয়েন্টেড ডাটাবেজ, আর্টিফিশয়াল ইন্টেলিজেন্স, নিউরাল নেটওয়ার্কস ও প্যারালাল প্রোগ্রামিং, অফিস অটোমেশন সিস্টেম, ক্যাড/ক্যাম এবং ইন্টারনেট ও ওয়েব সাইট ভিত্তিক প্রোগ্রাম রচনার ক্ষেত্রে জাভা সফলভাবে ব্যবহৃত হচ্ছে। তাই বলা যায় জাভা ভাষার ব্যবহারিক ক্ষেত্র বা প্রয়োগক্ষেত্র বিশাল ও ব্যাপক। তবে বিশাল ও ব্যাপক এই প্রয়োগক্ষেত্রকে মূলত তিনটি প্রধান ক্ষেত্র হিসেবে চিহ্নিত করা যায়। যথা ঃ

- কনসোল মোড অ্যাপ্লিকেশন (Console mode applicaion)
- গ্রাফিক্যাল ইউজার ইন্টারফেস অ্যাপ্লিকেশন (GUI application) এবং
- অ্যাপলেট (Applet) ইত্যাদি।

কনসোল মোড অ্যাপ্লিকেশন (Consol mode Application) ঃ Console mode application-কে বাহ্যিক প্রকাশের দিক থেকে DOS-এর সাথে তুলনা করা যায়। DOS প্রকৃতপক্ষে একটি Console mode Operating System । বিভিন্ন ধরনের টেক্সট ভিত্তিক প্রোগ্রাম Console মোডে রচনা করা হয়। এখানে গ্রাফিক্সের কোন স্থান নেই, সব কাজ করা হয় টেক্সট ব্যবহার করে। গ্রাফিক্স না থাকার ফলে এধরনের প্রোগ্রাম GUI প্রোগ্রামের তুলনায় দ্রুতগতিসম্পন্ন এবং এদের Overhead কম। সাধারণত সার্ভার সাইড প্রোগ্রাম তৈরিতে এ পদ্ধতি ব্যবহার করা হয়।

গ্রাফিক্যাল ইউজার ইন্টারফেস অ্যাপ্লিকেশন (GUI application) ঃ GUI এর পূর্ণরূপ হচ্ছে- Griphical User Interface. প্রতিটি Standard languageই GUI component প্রদান করে। তবে অন্যদের সাথে Java এর মূল পার্থক্য হচ্ছে portability। Java language দ্বারা তৈরি একটি graphical user interface যে কোন platform যেমন Sun workstation, IBM-compatible PC, Macintosh এবং অন্যান্য অনেক ধরনের platform- এ চলবে। Java AWT এই অনবদ্য ব্যাপারটিকে বাস্তব রূপ দিয়েছে।

Graphical user interface (GUI) কে একটি application এর চেহারা (interface) বলা যেতে পারে, যার মাধ্যমে application এর user বা application এর সাথে যোগাযোগ রক্ষা করে। তাই এর ইউজাররা খুব সহজে এটিকে ব্যবহার করতে পারে।

GUI applicaitonসমূহ আসলে কিছু GUI component এর সমষ্টিমাত্র। GUI component গুলো হচ্ছে এক ধরনের Object, আরো সুনির্দিষ্টভাবে বলতে গেলে visual বা graphical object এবং user-রা এই object গুলোর সাথে mouse বা keyboard দ্বারা যোগাযোগ করে। এই componentসমূহ যেমন- Button, Label, TextField ইত্যাদি componentসমূহ আমরা actually application অথবা applet এর screen এ দেখতে পারি। যে opertation বা method সমূহ প্রতিটি GUI component এ common বা সাধারণ সেই methodসমূহ superclass component এ বিদ্যমান। এই componentসমূহ যেসব class হতে তৈরি করা হবে সেই class গুলো Java.awt প্যাকেজের মধ্যে আছে। AWT মানে হল Abstract Windowing Toolkit। AWT package এর ভেতর প্রয়োজনীয় class, interface ও অন্যান্য packageসমূহ রয়েছে।

অ্যাপলেট (Applet) ঃ অ্যাপলেট জাভা ল্যাঙ্গুয়েজে রচিত একটি বিশেষ ধরনের প্রোগ্রাম। অন্যকোন ল্যাঙ্গুয়েজ এ অ্যাপলেট তৈরির কোন ব্যবস্থা নেই। কাজেই অ্যাপলেট জাভার একটি নিজস্ব ফিচার। এটি এইচ.টি.এম.এল (HTML) পেজ এ সংযুক্ত (embeded) অবস্থায় চলে। ব্রাউজারের মধ্যস্থ বিল্ট ইন জাভা ইন্টারপ্রিটার এর মাধ্যমে অ্যাপলেট রান করে। সুতরাং যে ব্রাউজারে বিল্ট ইন জাভা ইন্টারপ্রিটার নেই সেটিতে অ্যাপলেট রান করা সম্ভব নয়। তবে জনপ্রিয় সব ব্রাউজারে জাভা ইন্টারপ্রিটার আছে। যেমন– Netscape, Internet Explorer ইত্যাদি।

ইদানিং ওয়েব পেইজগুলোতে Dynamic HTML, scripting ইত্যাদি সন্নিবেশিত হওয়ায় ওয়েব পেইজগুলো আগের তুলনায় অনেক বেশি ক্ষমতা সম্পন্ন। কিন্তু তারপরও যেহেতু Applet পরিপূর্ণ প্রোগ্রামিং ল্যাংগুয়েজে লেখা কাজেই Applet এর ক্ষমতা HTML, XML, scripting ইত্যাদির তুলনায় অনেক অনেক গুণ বেশি। মূলত একটি পরিপূর্ণ প্রোগ্রামের মাধ্যমে যা কিছু করা সম্ভব Applet এও তা সম্ভব। তবে নিরাপত্তার কথা বিবেচনা করে Applet-এর উপর বেশ কিছু বিধি-নিষেধ আরোপ করা আছে, যা সাধারণ Program- এ নেই। যদিও Applet-এ চমৎকার Graphics ব্যবহারের সুবিধা আছে- তারপরেও শুধু ওয়েব পেইজের শ্রীবৃদ্ধির জন্য Applet ব্যবহৃত হয় না বরং Applet এর ব্যবহার একটি ওয়েব পেইজকে অনেক বেশি জীবন্ত বা Interactive করে তোলে।

৫.৫ জাভা ভাষার প্রোগ্রামিং স্টাইল ও প্রোগ্রাম লেখার নিয়মাবলি (Programming style & convention of Java) ঃ

প্রোগ্রামিং স্টাইল (Programming style) ঃ একটি জাভা প্রোগ্রামের নমুনা নিম্নরূপ ঃ

```
public class Myfirstjava
{
public static void main (String args[])
{
System.out.println("I love Bangladesh");
}
}
```

এখানে public class Myfirstjava দ্বারা class ঘোষণা করা হয়েছে। এই লাইনটি হচ্ছে class header. প্রতিটি জাভা class এর একটি class header এবং class body থাকে। এখানে class body শুরু হয়েছে ২নং লাইনে "{" (opening brace এ) এবং শেষ হয়েছে ৭ নং লাইনে "}" (closing brace) এর মাধ্যমে। এই দুই দ্বিতীয় বন্ধনীর মধ্যে যা আছে সবই class body'র অর্ন্তভুক্ত। যেহেতু জাভা একটি পরিপূর্ণ OOP ল্যাঙ্গুয়েজ, কাজেই এর সবকিছুই থাকবে class এর মধ্যে।

এই প্রোগ্রামের class header টির প্রথম শব্দ public। এটি একটি java keyword এবং একটি access specifier। public শব্দটির মাধ্যমে বুঝানো হয়েছে যে এই class টি অন্য সকল class এর কাছে উম্মুক্ত বা unprotected । class header এর দ্বিতীয় শব্দটি হচ্ছে class । ইহা একটি keyword এবং ইহার ঘোষণার ফলে পরবর্তী শব্দটি হবে class এর নাম এবং নতুন class definition । এখানে Myfirstjava একটি Java identifier যা class টির নাম হিসেবে ব্যবহৃত হচ্ছে।

জাভাতে স্পেস এবং পজিশন কোন ব্যাপার নয়, কাজেই জাভা প্রোগ্রামকে বিশাল লাইনেও লেখা যায় আবার প্রতিটি শব্দকে আলাদা লাইনেও লেখা যায়। যেভাবেই লেখা হোক না কেন কম্পাইলারের কাছে একই অর্থ বহন করে।

যেমন– উপরোক্ত প্রোগ্রামকে এভাবেও লেখা যায়–

```
public
class
Myfirstjava
{
public
static
void
main
(String
args[])
{
System.
out.
println("I love Bangladesh");
}
}
```

প্রোগ্রাম লেখার নিয়মাবলি (Programming convention) ঃ প্রোগ্রামের সৌন্দর্য ও readability বৃদ্ধির জন্য জাভা ভাষায় প্রোগ্রাম লেখার ক্ষেত্রে নিম্নবর্ণিত নিয়মাবলি মেনে চলা উচিত। যথা ঃ

- Class header এর শেষে বা পরের লাইনে Opening brace "{" বসানো উচিত।
- Opening brace এর "{" পরের লাইন থেকে এক ঘর Tab position ডান দিকে সরে গিয়ে Class body লেখা শুরু করা উচিত।
- একটি ব্লকের ভেতর আরেকটি ব্লক আসলে ভেতরের ব্লকটি আরো এক Tab position ডানে Indent করে লেখা উচিত।
- প্রতিটি Statement আলাদা লাইনে লিখা উচিত।

তাছাড়া প্রোগ্রামে ব্যবহৃত বিভিন্ন আইডেন্টিফায়ারের (যেমন- ClassName, Constant, MethodName, FieldName) নামকরণের ক্ষেত্রেও কিছু নিয়মাবলি আছে যেগুলো মেনে চললে প্রোগ্রামে ব্যবহৃত ক্লাস, কন্সট্যান্ট, মেথড ইত্যাদিকে পৃথকভাবে চিহ্নিত করা যায়। নিময়গুলো হচ্ছে ঃ

- Class name এর ক্ষেত্রে প্রতিটি শব্দের প্রথম অক্ষের বড় হাতের (Capitalized) এবং বাকী অক্ষরগুলো ছোট হাতের লেখা উচিত। যেমন-

  CapitalizedEveryWord

  বা,

  MyClass
- Constant এর ক্ষেত্রে সব অক্ষর বড় হাতের (Capital letter) লিখা উচিত। শব্দগুলোকে আলাদা করা হয় Underscore ব্যবহার করে। যেমন-

  CAPITALIZE_WITH_UNDERSCORE

  বা,

  MY_CONSTANT
- Method এবং field name এর ক্ষেত্রে প্রথম শব্দ ছোট হাতের অক্ষরে বাকি সব শব্দের প্রথম অক্ষর বড় হাতের অক্ষর লেখা উচিত। যেমন–

  startWithLowercaseAndCapitilizeOtherWords

  বা, myMethod

  বা, myField

৫.৬ জাভায় ব্যবহৃত ফাঁকা স্থান, আইডেন্টিফায়ার, লিটারেলস, কমেন্টস, সেপারেটর ও কীওয়ার্ড (White Space Identifier, Literals, Comments, Separators and Keyword) ঃ

ফাঁকা স্থান (White Space) ঃ কোন স্টেটমেন্টে ব্যবহৃত ভেরিয়েবল ও ডাটার মধ্যবর্তী ফাঁকা জায়গাকে white space বলে। Tabs, line-feeds এবং carriage return (Enter) ইত্যাদিও white space হিসেবে বিবেচিত হয়।

উদাহরণ ঃ

```
public   static   void   main (string args [ ])
       ↑        ↑      ↑
       └────────┴──────┘
          White Space
```

আইডেন্টিফায়ার (Identifier) ঃ প্রোগ্রামে ব্যবহৃত ভেরিয়েবল, কনস্ট্যান্ট, ক্লাস, অবজেক্ট, লেবেল, স্ট্রাকচার, পয়েন্টার, অ্যারে, মেথড ইত্যাদিকে দেয়া যে কোন বৈধ নামকে আইডেন্টিফায়ার বলে। আইডেন্টিফায়ারের নাম প্রোগ্রাম লেখার সময় দেয়া হয়। ভেরিয়েবলও এক প্রকার আইডেন্টিফায়ার। সুতরাং ভেরিয়েবল এবং আইডেন্টিফায়ার নামকরণের নিয়ম একই। একবার একটি আইডেন্টিফায়ার তৈরি হলে প্রোগ্রামের যেকোন জায়গায় সে একই অবজেক্টকে নির্দেশ করে।

যেমন ঃ int sum = x+y

এখানে sum একটি সঠিক আইডেন্টিফায়ার ।

সঠিক আইডেন্টিফায়ার তৈরির জন্য নিম্নবর্ণিত নিয়ম অনুসরণ করতে হয়-

- আইডেন্টিফায়ারের প্রথম স্থানটি অবশ্যই অক্ষর হতে হবে, পরের সংখ্যা অক্ষর বা সংখ্যা হতে পারে।
- জাভা ইউনিকোড (Unicode) এর উপর তৈরি বিধায়, স্ট্যান্ডার্ড আইডেন্টিফায়ার যেকোন ভাষায় হতে পারে উদাহরণস্বরূপ Arabic, Bengali, Tamil, Thai ইত্যাদি।
- আন্ডারস্কোর (_) এবং ডলার চিহ্ন ($) ছাড়া অন্য কোন বিশেষ অক্ষর (ইত্যাদি) ব্যবহার করা যাবে না। তবে আন্ডারস্কোর (_) এবং ডলার চিহ্ন ($) আইডেন্টিফায়ারের প্রথম অক্ষর হিসেবে ও ব্যবহার করা যাবে।
- সি এবং অন্যসব আধুনিক ল্যাংগুয়েজের মতো জাভায় আইডেন্টিফায়ার Case sensitive। এর অর্থ হলো SUM এবং sum দুটি ভিন্ন আইডেন্টিফায়ার।
- আইডেন্টিফায়ারের নাম ততটুকু বড় হওয়া প্রয়োজন যাতে সে তার নিজের বর্ণনা নামের মধ্যে প্রকাশ করতে পারে।
- আইডেন্টিফায়ারের নাম হিসেবে কোন কীওয়ার্ড (Keyword) ব্যবহার করা যাবে না।
- আইডেন্টিফায়ারের নামের মাঝে কোন ফাঁকা জায়গা (Space) থাকতে পারবেনা।

Legal Identifiers	Illegal identifiers
Helloworld	1helloworld
Counter	Counter & add
Hotjava$	Hot Java
ioc_Queve3	65536
TableTenisBat	Table-Tenis

লিটারেলস (Literals) ঃ লিটারেলস হলো এক ধরনের মান যা কোন ভেরিয়েবলে সংরক্ষণ করা হয় অথবা কোন মেথডের আরগুমেন্ট হিসেবে পাস করা হয়।

উদাহরণ ঃ

```
int sum = 0;
static string lastName = "Shovon";
borlean isBig = true;
```

জাভায় ৫ ধরনের লিটারেলস ব্যবহার হয়। যথা ঃ

- বুলিয়ান লিটারেলস (Boolean literals)
- ইন্টিজার লিটারেলস (Intger literals)
- ক্যারেক্টার লিটারেলস (Character literals)
- স্ট্রিং লিটারেলস (String literals) এবং
- ফ্লোট লিটারেলস (Float literals)

বুলিয়ান লিটারেলস (boolean literals) ঃ boolean type literal দুই ধরনের। যথা ঃ true or false

উদাহরণ ঃ

```
boolean male = true
boolean female = false
```

ইন্টিজার লিটারেলস (Intger literals) ঃ ইন্টিজার লিটারেল প্রকাশ করা হয় ডেসিম্যাল, অক্টাল এবং হেক্সাডেসিম্যালে। default মান হয় ডেসিম্যাল সংখ্যায়। অক্টালে প্রকাশ করার জন্য সর্বপ্রথমে লিটারেলটির 0 (Zero) থাকে। হেক্সাডেসিম্যালে প্রকাশ করার জন্য সর্বপ্রথমে ox অথবা OX থাকে এবং হেক্সাডিজিটগুলো বড় হাতের অথবা ছোট হাতের হতে পারে। উদাহরণস্বরূপ ২৯ এই মানটি ছয়ভাবে প্রকাশ করা যায়।

29	ডেসিম্যাল
o35	অক্ট্যাল
OXId	হেক্সাডেসিম্যাল
oXID	
OxId	
oxID	

ক্যারেক্টার লিটারেলস (Character literals) :

ক্যারেক্টার লিটারেল সবসময় Single Quotes (' ') এর মধ্যে থাকবে।

উদাহরণ ঃ

```
char Big = 'B'
```

অন্যভাবেও ক্যারেক্টার লিটারেলকে প্রকাশ করা যায় - যাতে ৪টি হেক্সাডেসিম্যাল ডিজিট থাকবে এবং ডিজিটের আগে \u থাকবে এবং সম্পূর্ণ লিটারেলটি Single quote এর মধ্যে আবদ্ধ থাকবে।

উদাহরণ ঃ `char smal= '\u4567' ;`

বিশেষ ক্যারেক্টারকে বুঝানোর জন্য জাভা কিছু escape sequence সাপোর্ট করে যা নিম্নে দেয়া হলো:

Escape literals	Meaning
'\b'	\u0008 back space
'\t'	\u0009 horizontal tab
'\n'	\u000a linefeed
'\f'	\u000c from feed
'\r'	\u000d carriage return
'\" '	\u0022 double quotation mark
'\' '	\u0027 single quotation mark
'\\'	\u005c backslash

টেবিল ঃ

স্ট্রিং লিটারেলস (String literals) ঃ জাভা প্রোগ্রামে যদি এক বা একাধিক ক্যারেক্টার ডাবল কোটেশন (" ") মধ্যে রাখা হয় তবে তাকে স্ট্রিং বলা হয়।

উদাহরণ ঃ

```
string s="Hello World";
```

কম্পাইলার স্ট্রিং কে কনস্ট্যান্ট হিসেবে নেয়। উল্লেখ্য, স্ট্রিং-এর সবগুলো ক্যারেক্টার এক লাইনে রাখতে হয়।

স্ট্রিং লিটারেলের মধ্যে escape sequence থাকতে পারে যা character literals-এ আলোচনা করা হয়েছে। যেমন–

```
string sr="java \n matic";
```

ফ্লোটিং পয়েন্ট লিটারেলস (Floating point literals) ঃ ফ্লোটিং পয়েন্ট লিটারেল দশমিকযুক্ত সংখ্যা প্রকাশ করে। ফ্লোটিং পয়েন্ট লিটারেলকে প্রকাশ করতে হলে নিচের যে কোন একটি পদ্ধতি অবলম্বন করতে হবে। যেমন–

- দশমিকযুক্ত সংখ্যা : 1.414
- বৈজ্ঞানিক সংখ্যার জন্য : 4.23E +21 অথবা 4.23e+21
- ৩২ বিট ফ্লোট লিটারেল : 1.828f অথবা 1.828F
- ৬৪ বিট ডাবল লিটারেল ; 1234d অথবা 1234D

কমেন্টস (Comments) ঃ প্রোগ্রামে অনেক সময় পরবর্তী সময়ে নিজে কিছু বুঝার জন্য কিংবা অন্যকে কিছু বুঝানোর জন্য কমেন্টস বা মন্তব্য লিখে রাখা হয়। প্রোগ্রাম নির্বাহে কমেন্টের কোন ভূমিকা নেই। সিঙ্গেল লাইন কমেন্টে ন্যূনতম দুটি স্লাশ (//) ব্যবহৃত হয়। অপরদিকে মাল্টি লাইন কমেন্ট '/*' দিয়ে শুরু হয় এবং '*/' দিয়ে শেষ হয়। '/*' এবং '*/' এর মধ্যবর্তী লাইনসমূহ কমেন্ট হিসেবে বিবেচিত হবে।

উদাহরণ ঃ

```
sum = sum+x;// This is a single line comment
sun = sum + x; /* This program is used to find out
                  the summation of a series */
```

সেপারেটর (Separator) ঃ সেপারেটর হিসাবে ব্যাক স্লাশ (\) ক্যারেক্টার ব্যবহৃত হয়। সাধারণত ফাইল বা ডিরেক্টরির পাথ (path) নির্দেশ করার জন্য সেপারেটর ব্যবহার করা হয়।

উদাহরণ ঃ

```
C : \jdk 1.3\bin
D : \ soft\Turboc\bin ইত্যাদি।
```

কীওয়ার্ড (Keyword) ঃ কীওয়ার্ড হল প্রোগ্রামের জন্য সংরক্ষিত বিশেষ শব্দ যা কম্পাইলার সনাক্ত করতে পারে এবং তদনুযায়ী কাজ করতে পারে। প্রত্যেক ল্যাংগুয়েজেই কিছু নির্দিষ্ট কীওয়ার্ড আছে যা ঐ ল্যাংগুয়েজে বিশেষ অর্থ বহন করে। প্রোগ্রামে ঐ শব্দগুলো নিদিষ্ট কাজে ব্যবহার না করলে কম্পাইলার এরর দেখাবে। কীওয়ার্ড ব্যবহারের জন্য সুনিদিষ্ট নিয়ম আছে। জাভায়ও এ ধরনের কিছু কীওয়ার্ড রয়েছে যাদেরকে English ল্যাংগুয়েজের verb, adjective এবং pronoun এর সাথে তুলনা করতে পারি। জাভা ভাষায় মোট ৬১টি কীওয়ার্ড ব্যবহৃত হয়। তন্মধ্যে ভবিষ্যতে ব্যবহারের জন্য সংরক্ষিত কীওয়ার্ডগুলোকে Bold আকারে দেখানো হল ঃ

জাভা কীওয়ার্ডস (Java Kewords)				
abstract	do	implements	private	throws
boolean	double	import	protected	transient
break	else	**inner**	public	true
byte	extends	instanceof	**rest**	try
byvalue	false	int	return	**var**
case	final	interface	short	void
cast	finally	long	static	volatile
catch	float	native	strictfp	while
char	for	new	super	widefp
class	**future**	null	switch	
const	**generic**	**operator**	synchronized	
continue	**goto**	**outer**	this	
default	if	package	throw	

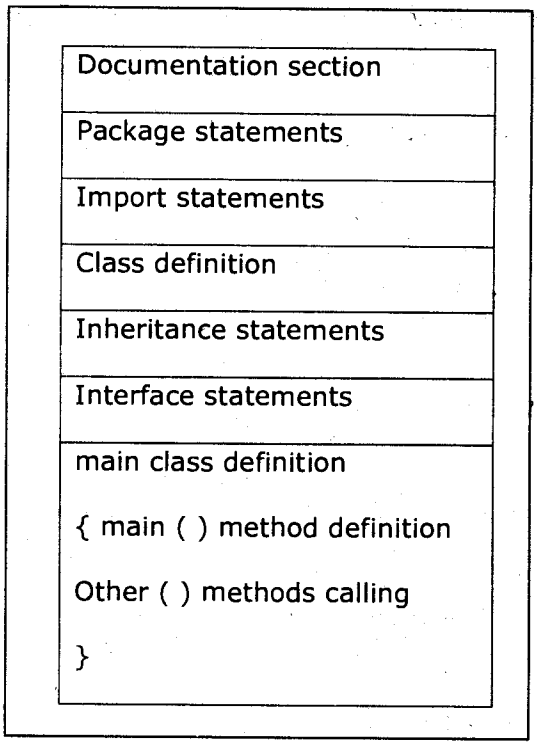
টেবিল ঃ জাভা কীওয়ার্ডস

৫.৭ জাভা প্রোগ্রামের স্ট্রাকচার (Structure of Java Program) ঃ

একটি জাভা প্রোগ্রামে এক বা একাধিক ক্লাস এবং প্রতিটি ক্লাসে এক বা একাধিক মেম্বার ভেরিয়েবল ও মেম্বার মেথড (ফাংশন) থাকে। জাভা প্রোগ্রাম রচনার সময় আমরা প্রথমে ক্লাসগুলোকে ডিফাইন (define) করে থাকি। তবে প্রতিটি জাভা প্রোগ্রামে অবশ্যই main ( ) নামে একটি পাবলিক (public) প্রকৃতির স্ট্যাটিক (static) মেথড থাকতে হবে। কারণ main ( ) মেথডের মধ্যেই অন্যান্য সকল মেথড ও অবজেক্টকে কল (call) করা হয়। একটি জাভা প্রোগ্রামকে main ( ) মেথডসহ অনেকগুলো সেকশনে ভাগ করা হয়। যেমন-

- ডকুমেন্টেশন সেকশন (Documentation section)
- প্যাকেজ স্টেটমেন্টস (Package statements)
- ইমপোর্ট স্টেসমেন্টস (Import statements)
- ক্লাস ডেফিনিশন (Class definition)
- ইনহেরিটেন্স স্টেটমেন্টস (Inheritance statements)
- ইন্টারফেস স্টেটমেন্টস (Interface statements)
- main ( ) ক্লাস ডেফিনিশন (main class definition)
- main ( ) মেথড ডেফিনিশন (main ( ) method definition)
- অন্যান্য মেথড কলিং (other methods calling) ইত্যাদি।

এসব সেকশনের মধ্যে কিছু সেকশন অবশ্যই প্রয়োজনীয় (essential) এবং কিছু সেকশন ঐচ্ছিক (Optional)।



চিত্র ঃ জাভা প্রোগ্রাম স্ট্রাকচার

ডকুমেন্টেশন সেকশন (Documentation section) ঃ এ সেকশনে প্রোগ্রাম সম্পর্কিত বিভিন্ন মন্তব্য লেখা হয়। যেমন-

```
// This is my first program in Java
```

প্যাকেজ স্টেটমেন্টস (Package statements) ঃ এই সেকশনটি ঐচ্ছিক। প্যাকেজ (Package) ডিক্লারেশনের মাধ্যমে প্রোগ্রামে ব্যবহৃত ইন্টারফেস এবং ক্লাসগুলোকে organize অবস্থায় ছোট ছোট ফোল্ডারের মাধ্যমে সাজিয়ে রাখা যায়। ফলে পরবর্তীতে খুব সহজেই নির্দিষ্ট প্যাকেজে গিয়ে নির্দিষ্ট ক্লাসটি খুঁজে পাওয়া যায়। Package কে নিম্নরূপে ডিক্লেয়ার করা যায়।

```
package mypackage;
```

এখানে 'package' হচ্ছে কীওয়ার্ড এবং 'mypackage' হচ্ছে প্যাকেজের নাম।

ইমপোর্ট স্টেটমেন্টস (Import statements) ঃ কোন লাইব্রেরি ক্লাস (Library class) কে প্রোগ্রামে ব্যবহার করতে চাইলে সে ক্লাসটি বা ক্লাসগুলো যে প্যাকেজে আছে তা import স্টেটমেন্টের মাধ্যমে ডিক্লেয়ার করা হয়। যেমন-

```
import java.awt *
```

অথবা,

```
import java.awt.label;
```

প্রথম স্টেটমেন্টের মাধ্যমে 'awt' প্যাকেজের সকল ক্লাসকে import করা হয় এবং দ্বিতীয় স্টেটমেন্টের মাধ্যমে 'awt' প্যাকেজের 'label' নামক ক্লাসকে import করা হয়। এটিও ঐচ্ছিক অংশ।

ক্লাস ডেফিনিশন (Class definition) ঃ একটি জাভা প্রোগ্রামে এক বা একাধিক ক্লাস থাকতে পারে। ক্লাস (Class) জাভা প্রোগ্রামের প্রাথমিক এবং অত্যাবশ্যকীয় অংশ। প্রোগ্রামে ব্যবহৃত অবজেক্টগুলো নির্মাণের জন্য class ব্যবহৃত হয়। ক্লাস ডিফাইন করার নিয়ম হল ঃ

```
public class Myclass
    {
    //class body
    }
```

এখানে 'Myclass' হল ক্লাসের নাম।

ইনহেরিটেন্স স্টেটমেন্টস (Inheritance statements) ঃ কোন একটি ক্লাসকে অন্য কোন ক্লাস দ্বারা সরাসরি ইনহেরিট করার জন্য ইনহেরিটেন্স পদ্ধতি ব্যবহার করা হয়। ইনহেরিটেন্স স্টেটমেন্ট হিসাবে 'extends' কী ওয়ার্ড ব্যবহার করা হয়। একটি ক্লাসকে অন্য কোন ক্লাস কর্তৃক ইনহেরিট করার নিয়ম হল ঃ

```
class A extends B
```

এখানে ক্লাস 'A' ক্লাস 'B' এর বৈশিষ্ট্য ইনহেরিট করছে।

এই সেকশনটি ঐচ্ছিক।

ইন্টারফেস স্টেটমেন্টস (Interface statements) ঃ মাল্টিপল ইনহেরিটেন্স পদ্ধতিকে জাভা ভাষায় ইন্টারফেস বলে। অর্থাৎ যখন কোন ক্লাস অন্য একাধিক ক্লাসকে ইনহেরিট করে তখন এই পদ্ধতিকে ইন্টারফেস বলে। ইন্টারফেস পদ্ধতিতে 'extends' এবং 'implements' এই দুটি কীওয়ার্ড ব্যবহৃত হয়। যেমন-

```
Class A extends B implements C
```

ইন্টারফেস স্টেটমেন্ট সেকশনটিও ঐচ্ছিক।

main ক্লাস ও main ( ) মেথড (main class and main ( ) method) ঃ কোন জাভা প্রোগ্রাম একাধিক ক্লাসের সমন্বয়ে তৈরি হলে Top level একটি ক্লাসে main ( ) মেথড অবশ্যই থাকতে হবে। কারণ প্রতিটি জাভা প্রোগ্রামই main ( ) মেথড invocation এর মাধ্যমে শুরু হয়। যেহেতু কোন অবজেক্ট তৈরি হওয়ার আগেই এই মেথডটি invoke হয় তাই এটি অবশ্যই static মেথড এবং finalও বটে। main ( ) মেথড ডিক্লারেশনের নিয়ম নিম্নরূপ ঃ

```
public static void main (string args [ ])
```

প্রোগ্রামে ব্যবহৃত অন্যান্য মেথড ও অবজেক্টসমূহ এই সেকশনেই কল (call) করতে হয়।

অনুশীলনী-৫

অতি সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। কে এবং কখন জাভা ভাষা ডেভেলপ করেন?

উত্তরঃ 'Sun Microsystems' কোম্পানির গবেষক James Gosling এর নেতৃত্বে একদল প্রোগ্রামার Green প্রজেক্টের আওতায় ১৯৯৬ সালে জাভা ভাষার সর্বপ্রথম ভার্সন প্রকাশ করেন।

২। JDK কী?

উত্তরঃ JDK এর পূর্ণ নাম Java Development Kit। প্রোগ্রাম রচনার জন্য প্রয়োজনীয় টুল এবং সহস্রাধিক ক্লাস ও মেথডসমূহ যে সিস্টেমের মধ্যে থাকে তাকেই JDK বলে।

৩। জাভার প্রয়োগক্ষেত্র উল্লেখ কর।

উত্তরঃ **জাভা ভাষার মূলতঃ** তিনটি প্রধান প্রয়োগক্ষেত্র রয়েছে। যথা ঃ

- কনসোল মোড অ্যাপ্লিকেশন এবং
- গ্রাফিক্যাল ইউজার ইন্টারফেস অ্যাপ্লিকেশন
- অ্যাপলেট অ্যাপ্লিকেশন।

৪। Identifier কী?

উত্তরঃ প্রোগ্রামে ব্যবহৃত Variable, constant, class, object, label, structure, pointer, array, method ইত্যাদিকে Identify করার জন্য যে নাম বা শব্দ ব্যবহার করা হয় তাকে Identifier বলে।

৫। Keyword কী? জাভায় মোট কতগুলো Keyword ব্যবহৃত হয়?

উত্তরঃ Keyword হচ্ছে প্রোগ্রামের জন্য সংরক্ষিত কিছু শব্দ। কম্পাইলারের কাছে যেগুলো বিশেষ অর্থ বহন করে। যেমন- class, int, char ইত্যাদি। জাভায় মোট ৬১টি Keyword ব্যবহৃত হয়।

৬। জাভায় ব্যবহৃত 10টি Keyword এর নাম লেখ।

উত্তরঃ জাভায় ব্যবহৃত 10টি Keyword নিম্নরূপ-

break, byte, case, char, do, else, go to, null, return, switch.

৭। মেইন মেথড [main () method] কী?

উত্তরঃ প্রতিটি জাভা প্রোগ্রাম শুরু হয় main () method এর মাধ্যমে। main () method বিভিন্ন Class এর Object তৈরি করে এবং এদের মধ্যে Communication এর ব্যবস্থা করে। এর Declaration নিম্নরূপ-

```
public static void main (String args[ ]){}
```

৮। JVM বলতে কী বোঝায়?

উত্তরঃ Java Virtual Machine কে সংক্ষেপে JVM বলে। এটি একটি কাল্পনিক Machine, যা Computer এর Memory তে অবস্থান করে Byte code তৈরি করে।

৯। কনসোল মোড কী কাজে বা কখন ব্যবহৃত হয়?

উত্তরঃ সাধারণত টেক্সটভিত্তিক প্রোগ্রাম রচনার ক্ষেত্রে কনসোল মোড ব্যবহৃত হয়।

১০। GUI মোড কী?

উত্তরঃ জাভাতে গ্রাফিক্সভিত্তিক অর্থাৎ উইন্ডোজভিত্তিক অ্যাপ্লিকেশন তৈরিতে যে মোডে প্রোগ্রাম রচনা করা হয় তাই GUI মোড।

১১। GUI মোডের জন্য প্রয়োজনীয় প্যাকেজটির নাম কী?

উত্তরঃ GUI মোডের জন্য অত্যাবশ্যকীয় প্যাকেজটি হচ্ছে AWT (Abstract windowing Toolkit) java. awt প্যাকেজ ইমপোর্ট করে GUI mode এর কাজগুলো করা যায়।

১২। লিটারেলস কী? জাভায় কয় ধরনের লিটারেলস ব্যবহৃত হয়?

উত্তরঃ লিটারেলস হলো এমন এক ধরনের মান যা কোন ভেরিয়েবলে সংরক্ষণ করা হয় অথবা কোন মেথডের আরগুমেন্ট হিসেবে পাস করা হয়।

জাভায় পাঁচ ধরনের লিটারেলস ব্যবহৃত হয়। যথা ঃ

- বুলিয়ান লিটারেলস,
- ইন্টিজার লিটারেলস,
- ক্যারেক্টার লিটারেলস,
- স্ট্রিং লিটারেলস এবং
- ফ্লোটিং পয়েন্ট লিটারেলস।

১৩। জাভায় সিঙ্গেল লাইন কমেন্টস ও মাল্টি লাইন কমেন্টস লেখার নিয়ম কী?

উত্তরঃ জাভায় সিঙ্গেল লাইন কমেন্টস লেখার জন্য দুটি স্লাশ (//) এবং মাল্টি লাইন কমেন্টস লেখার ক্ষেত্রে কমেন্টস '/*' দিয়ে শুরু হয় এবং '*/' দিয়ে শেষ হয়।

১৪। সেপারেটর (\) ব্যবহার করা হয় কেন?

উত্তরঃ সাধারণত ফাইল বা ডিরেক্টরির পাথ (Path) নির্দেশ করার জন্য সেপারেটর (\) ব্যবহার করা হয়।

➥ *সংক্ষিপ্ত প্রশ্নোত্তর ঃ*

১। জাভার বৈশিষ্ট্যগুলো কী কী?

উত্তর সংকেত ঃ ৫.১.১নং অনুচ্ছেদ দ্রষ্টব্য।

২। জাভা ভাষার প্রোগ্রামিং স্টাইল বর্ণনা কর।

উত্তর সংকেত ঃ ৫.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। জাভা প্রোগ্রামের স্ট্রাকচার লেখ।

উত্তর সংকেত ঃ ৫.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। জাভা সি++ এর ইন্টারনেট ভার্সন ব্যাখ্যা কর।

উত্তর সংকেত ঃ ৫.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। জাভা ও সি++ এর মধ্যে পার্থক্য লেখ।

উত্তর সংকেত ঃ ৫.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। **জাভাতে মূলতঃ কত ধরনের প্রোগ্রাম তৈরি করা যায়? সংক্ষেপে বর্ণনা কর।**

উত্তর সংকেত ঃ ৫.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৭। **জাভায় প্রোগ্রাম লেখার নিয়মাবলি কী কী?**

উত্তর সংকেত ঃ ৫.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৮। **জাভাকে যন্ত্র নিরপেক্ষ প্রোগ্রাম ভাষা বলা হয় কেন?**

উত্তর সংকেত ঃ ৫.১.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৯। **জাভা ভার্চুয়াল মেশিন (JVM) ব্যবহার করা হয় কেন? ব্যাখ্যা কর।**

উত্তর সংকেত ঃ ৫.২.১ নং অনুচ্ছেদ দ্রষ্টব্য।

১০। **জাভাকে পূর্ণাঙ্গ অবজেক্ট অরিয়েন্টেড ল্যাংগুয়েজ বলা হয় কেন?**

উত্তর সংকেত ঃ ৫.১.১ নং অনুচ্ছেদ দ্রষ্টব্য।

১১। **জাভাকে মাল্টিথ্রেডেড প্রোগ্রামিং ল্যাংগুয়েজে বলা হয় কেন?**

উত্তর সংকেত ঃ ৫.১.১ নং অনুচ্ছেদ দ্রষ্টব্য।

১২। **JDK তে বিদ্যমান টুলস ও তাদের ব্যবহার উল্লেখ কর।**

উত্তর সংকেত ঃ ৫.২ নং অনুচ্ছেদ দ্রষ্টব্য।

➥ *রচনামূলক প্রশ্নোত্তর ঃ*

১। **জাভা ভাষার ইতিহাস বর্ণনা কর।**

উত্তর সংকেত ঃ ৫.১ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **জাভার বৈশিষ্ট্যগুলো বর্ণনা কর।**

উত্তর সংকেত ঃ ৫.১.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **জাভা ডেভেলপমেন্ট এনভায়রনমেন্ট এর ধাপসমূহ বর্ণনা কর।**

উত্তর সংকেত ঃ ৫.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। **জাভা প্রোগ্রামের স্ট্রাকচার বা সংগঠন বর্ণনা কর।**

উত্তর সংকেত ঃ ৫.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

অধ্যায় ০৬

ডাটা টাইপ, ভেরিয়েবল, অপারেটর কন্ট্রোল স্টেটমেন্ট ও অ্যারে
Data Types, Variable, Operators, Control Statements & Array

৬.০ ভূমিকা (Introduction) ঃ

প্রোগ্রামে সাধারণত ডাটা নিয়ে কাজ করা হয়। প্রোগ্রামে ব্যবহারের জন্য ডাটাকে প্রথমে মেমোরিতে সংরক্ষণ করা হয় এবং প্রয়োজনে মেমোরি থেকে ডাটা উত্তোলন করে কাজে লাগান হয়। নিম্ন পর্যায়ের ভাষায় মেমোরিতে ডাটা রাখার জন্য সরাসরি বিট, বাইট এবং মেমোরি অ্যাড্রেস ব্যবহার করা হয়, যা বড় বড় প্রোগ্রামের জন্য জটিল এবং কষ্টকর। কারণ লক্ষ লক্ষ অ্যাড্রেসের মধ্যে কখন কোন অ্যাড্রেসে কোন ডাটা রাখা হল তা মনে রাখা অসম্ভব। এই অসুবিধা দূর করার জন্য এবং প্রোগ্রামকে সহজ করার লক্ষ্যে হাই লেভেল ল্যাংগুয়েজে বিট, বাইট ও মেমোরি অ্যাড্রেসের পরিবর্তে ভেরিয়েবল ব্যবহার করা হয়। ভেরিয়েবল হল প্রোগ্রামার কর্তৃক দেয়া কয়েকটি বিট বা বাইট সংরক্ষণের জন্য মেমোরি স্পেসের নাম, যে নামের অধীনে ডাটা রাখা হয়। এ ক্ষেত্রে প্রোগ্রামার বা প্রোগ্রাম ব্যবহারকারীর জানার দরকার নেই যে, মেমোরির কোন অ্যাড্রেসে কোন ডাটা রাখা হয়। কেবল সঠিক নিয়মে উপযুক্ত ডাটা টাইপসহ প্রয়োজন মত ভেরিয়েবল ঘোষণা করে ডাটা রাখা যায় এবং প্রয়োজনে তা সংরক্ষণ করে পরবর্তীতে ব্যবহার করা যায়। উল্লেখ্য, কোন ভেরিয়েবলে যে কোন সময় কেবল একটি মাত্র ডাটা রাখা সম্ভব। ভেরিয়েবলের পুরাতন মান সর্বদাই নতুন মান দ্বারা প্রতিস্থাপিত হয়।

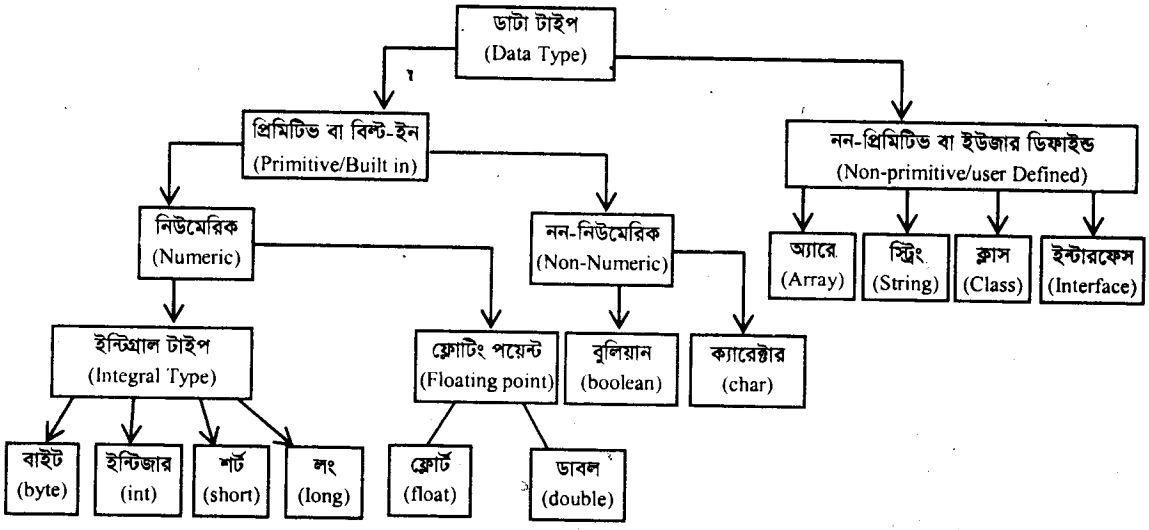
অন্যান্য প্রোগ্রামের মত প্রতিটি জাভা প্রোগ্রাম এক গুচ্ছ এক্সপ্রেশন বা স্টেটমেন্টের সমন্বয়ে গঠিত। প্রতিটি এক্সপ্রেশন আবার কতগুলো প্রতীক, সংকেত, ভেরিয়েবল তথা টোকেন, কীওয়ার্ড, অপারেটর, অপারেন্ড ইত্যাদির সমন্বয়ে গঠিত। এদের কোনটি প্রোগ্রামে ব্যবহৃত ডাটা ধারণ করে, কোনটি ডাটার গাণিতিক বা যৌক্তিক ক্রিয়া সম্পন্ন করে, কোনটি প্রোগ্রামে ব্যবহৃত অন্যান্য স্টেটমেন্ট নিয়ন্ত্রণ করে, আবার কোনটি বিশেষ প্রোগ্রাম কোড উৎপন্ন করে। এই অধ্যায়ে ডাটা টাইপ, ভেরিয়েবল, অ্যারে, অপারেটর, অপারেন্ড, এক্সপ্রেশন এবং কন্ট্রোল স্টেটমেন্ট সহ বিস্তারিত আলোচনা করা হয়েছে।

৬.১ জাভা ভাষায় ডাটা টাইপ (Data Types of Java Programs) ঃ

প্রোগ্রামে ব্যবহৃত যে কোন মানই (value) ডাটা। জাভা ভাষায় ইন্টিজার, ক্যারেক্টার, ফ্লোটিং পয়েন্ট, স্ট্রিং, লজিকাল ইত্যাদি প্রায় সব ধরনের ডাটা নিয়েই কাজ করা যায়। তবে ডাটার মান, ধরন এবং মেমোরি স্পেস সংরক্ষণের ভিত্তিতে জাভা ভাষায় ব্যবহৃত ডাটাসমূহকে প্রধানত দুটি ভাগে ভাগ করা যায়। যথা ঃ

- প্রিমিটিভ বা বিল্ট-ইন ডাটা টাইপ (Primitive/Built-in Data Type)
- নন প্রিমিটিভ বা ইউজার ডিফাইন্ড ডাটা টাইপ (Non primitive/user defined Data Type)।

নিম্নে জাভায় ব্যবহৃত ডাটা টাইপের একটি হায়ারারকি উল্লেখ করা হল।



চিত্র ঃ জাভা ডাটা টাইপ

প্রিমিটিভ ডাটা টাইপ গুলোকে নন-নিউমেরিক এবং নিউমেরিক এই দুই টাইপে ভাগ করা হয়। নন-নিউমেরিক ডাটাকে আবার বুলিয়ান ও ক্যারেক্টার এবং নিউমেরিক ডাটাকে পূর্ণমান ও ভগ্নাংশ টাইপে ভাগ করা হয়। জাভা ভাষায় বুলিয়ান ডাটা ব্যবহারের জন্য boolean টাইপ, ক্যারেক্টার ডাটা ব্যবহরের জন্য char টাইপ, পূর্ণমান ডাটা ব্যবহারের জন্য byte, short, int ও long টাইপ এবং ভগ্নাংশ ডাটা ব্যবহারের জন্য float ও double টাইপ ব্যবহৃত হয়। নন-প্রিমিটিভ ডাটা টাইপকে কাস্টম বা ইউজার ডিফাইন্ড ডাটা টাইপ বলা হয়, যেগুলো মূলত এক বা একাধিক প্রিমিটিভ ডাটা টাইপের সমন্বয়ে গঠিত হয়। বিভিন্ন প্রকার প্রিমিটিভ ডাটা টাইপ ভেরিয়েবলের জন্য সংরক্ষিত মেমোরি পরিসরের পরিমাণ এবং ডাটার রেঞ্জ বা ব্যাপ্তি উল্লেখ করা হল।

ডাটা টাইপ	বিট সংখ্যা	ডাটার রেঞ্জ বা ব্যাপ্তি
boolean	1	false বা true
char	16	0 থেকে 65535 ইউনিকোড ক্যারেক্টার
byte	8	-128 থেকে 127 বা -2^7 থেকে 2^7-1
short	16	-32768 থেকে 32767 বা, -2^{15} থেকে $2^{15}-1$
int	32	-2147483648 থেকে 21474883647 বা, -2^{31} থেকে $2^{31}-1$
long	64	-9223372036854775808 থেকে 9223372036854775807 বা, -2^{63} থেকে $2^{63}-1$
float	32	$\pm$ 3.4 x 10^{-38}থেকে $\pm$ 3.4 x 10^{38}
double	64	$\pm$ 1.7 x 10^{-308} থেকে $\pm$ 1.7 x 10^{308}

নিম্নে প্রিমিটিভ বা বিল্ট-ইন ডাটা টাইপসমূহ আলোচনা করা হল ঃ

বুলিয়ান টাইপ (boolean) ঃ জাভা ভাষায় লজিক্যাল ডাটা (অর্থাৎ সত্য কিংবা মিথ্যা) নিয়ে কাজ করার জন্য boolean ডাটা টাইপ ব্যবহৃত হয়। অর্থাৎ লজিক্যাল টাইপ ভেরিয়েবল ঘোষণার জন্য boolean কীওয়ার্ড ব্যবহৃত হয়।

উদাহরণ ঃ boolean b;

উপরোক্ত স্টেটমেন্টের মাধ্যমে b নামের boolean টাইপের কোন ভেরিয়েবল ঘোষণা করলে কম্পাইলার প্রোগ্রাম নির্বাহ কালে মেমোরিতে b নামে এক বিট জায়গা সংরক্ষণ করে। boolean টাইপ ভেরিয়েবলের মান হিসেবে কেবল true অথবা false নির্ধারণ করা যেতে পারে। true এবং false দুইটি জাভা কীওয়ার্ড যেগুলোর মান যথাক্রমে ১ এবং ০ হিসেবে গণ্য হয়। সুতরাং এদের অন্য কোনভাবে লেখা যায় না।

ক্যারেক্টার টাইপ (char) ঃ জাভা ভাষায় ক্যারেক্টার টাইপ ডাটা (অর্থাৎ বিভিন্ন অক্ষর ও প্রতীক) নিয়ে কাজ করার জন্য ক্যারেক্টার টাইপ ভেরিয়েবল ব্যবহার করা হয়। ক্যারেক্টার টাইপ ভেরিয়েবল ঘোষণা করার জন্য char কীওয়ার্ড ব্যবহৃত হয়। সি/সি++ ক্যারেক্টার টাইপ ভেরিয়েবল অপেক্ষা জাভা ক্যারেক্টার টাইপ ভেরিয়েবলের পার্থক্য হল সি/সি++ ক্যারেক্টার টাইপ ভেরিয়েবলে ক্যারেক্টার ছাড়াও পূর্ণ সংখ্যা নিয়ে কাজ করা যায়। কিন্তু জাভা ক্যারেক্টার টাইপ ভেরিয়েবলে ক্যারেক্টার ছাড়া অন্য কোন টাইপ ডাটা নিয়ে কাজ করা যায় না। সি/সি++ ক্যারেক্টার টাইপ ভেরিয়েবল মেমোরিতে ১ বাইট জায়গা নেয়। কিন্তু জাভা ক্যারেক্টার টাইপ ভেরিয়েবল মেমোরিতে ২ বাইট বা ১৬ বিট জায়গা সংরক্ষণ করে।

জাভা ভাষায় char টাইপ ভেরিয়েবল মূলত ASCII ক্যারেক্টার জন্য ব্যবহৃত হয়। প্রোগ্রামে সাধারণত দুইভাবে char টাইপ ভেরিয়েবলের মান দেয়া যায়ঃ ক্যারেক্টারের ASCII মান ব্যবহার করে , অথবা সরাসরি অক্ষর ব্যবহার করে। তবে দ্বিতীয়ক্ষেত্রে ক্যারেক্টারটিকে সিঙ্গেল (' ') মধ্যে রাখতে হয়।

উদাহরণ ঃ

```
char ch1= 'A' ;
char ch2=65 ;
```

বাইট টাইপ (byte) ঃ

জাভা ভাষায় নেটওয়ার্ক সিস্টেম কিংবা ডাটা ফাইল হ্যান্ডলিং-এ ডাটা স্ট্রিম নিয়ে কাজ করার জন্য বাইট টাইপ ভেরিয়েবল ব্যবহৃত হয়। আবার সরাসরি বাইনারি ডাটা নিয়ে কাজ করার জন্যও বাইট টাইপ ভেরিয়েবল ব্যবহৃত হয়। বাইট টাইপ ভেরিয়েবলের জন্য কম্পাইলার মেমোরিতে এক বাইট বা ৮ বিট জায়গা সংরক্ষণ করে। এই ৮ বিট জায়গার মধ্যে ডান দিক থেকে প্রথম ৭ টি বিট (বিট ০ থেকে বিট ৬) ডাটা রাখার জন্য এবং সবচেয়ে বামের বিটটি (বিট-৭) সাইন বিট (ধনাত্মক না ঋণাত্মক) হিসেবে ব্যবহৃত হয়। ধনাত্মক ডাটার জন্য সাইন বিটটি শূন্য (০) হয় এবং ঋণাত্মক ডাটার জন্য সাইন বিটটি এক (১) হয়।

বাইট টাইপ ভেরিয়েবল ঘোষণা জন্য byte কীওয়ার্ড ব্যবহৃত হয়।

উদাহরণ ঃ

```
byte Value1=65;
```

স্টেটমেন্টের মাধ্যমে Value1 নামে byte টাইপের কোন ভেরিয়েবল ঘোষণা করলে কম্পাইলার প্রোগ্রাম নির্বাহকালে মেমোরিতে Value1 নামে এক বাইট জায়গা বরাদ্দ করে সেখানে ৬৫ সংরক্ষণ করবে।

শর্ট টাইপ (short) ঃ

জাভা ভাষা -৩২৭৬৮ থেকে ৩২৭৬৭ এর মধ্যবর্তী যে কোন পূর্ণ সংখ্যা (যেমন ২৩, -৪৫৬, ২৩৪৫ ইত্যাদি) নিয়ে কাজ করার জন্য short টাইপ ভেরিয়েবল ব্যবহৃত হয়। short টাইপ ভেরিয়েবল ঘোষণার জন্য short কীওয়ার্ড ব্যবহৃত হয়। প্রতিটি short টাইপ ভেরিয়েবলের জন্য কম্পাইলার মেমোরিতে ২ বাইট বা ১৬ বিট জায়গা সংরক্ষণ করে। ১৬ বিট জায়গার মধ্যে ডান দিক থেকে প্রথম ১৫ টি বিট (বিট ০ থেকে বিট ১৪) ডাটা রাখার জন্য এবং বামের বিটটি (বিট-১৫) সাইন বিট হিসেবে ব্যবহৃত হয়। ধনাত্মক ডাটার জন্য সাইন বিটটি শূন্য (০) হয় এবং ঋণাত্মক ডাটার জন্য সাইন বিটটি এক (১) হয়।

উদাহরণ ঃ

```
short Value1 =65;
```

স্টেটমেন্টের মাধ্যমে Value1 নামের কোন short টাইপের ভেরিয়েবল ঘোষণা করলে কম্পাইলার প্রোগ্রাম নির্বাহের সময় মেমোরিতে Value1 নামে ২ বাইট জায়গা বরাদ্দ করে সেখানে ৬৫ সংরক্ষণ করে।

ইন্টিজার টাইপ (int) ঃ

জাভা ভাষায় -২১৪৭৪৮৩৬৪৮ থেকে ২১৪৭৪৮৩৬৪৭ এর মধ্যবর্তী যে কোন পূর্ণসংখ্যা যেমন– ২৩, - ৪৭৮, ২৩৪৫ ইত্যাদি নিয়ে কাজ করার জন্য int টাইপ ভেরিয়েবল ব্যবহৃত হয়। পূর্ণসংখ্যা নিয়ে কাজ করার জন্য int টাইপই সবচেয়ে বেশি ব্যবহৃত হয়। int টাইপ ভেরিয়েবল ঘোষণার জন্য int কীওয়ার্ড ব্যবহৃত হয়। প্রতিটি int টাইপ ভেরিয়েবলের জন্য কম্পাইলার মেমোরিতে ৪ বাইট বা ৩২ বিট জায়গা সংরক্ষণ করে। ৩২ বিট জায়গার মধ্যে ডান দিক থেকে প্রথম ৩১ টি বিট (বিট ০ থেকে বিট ৩০) ডাটা রাখার জন্য এবং বামের বিটটি (বিট ৩১) সাইন বিট হিসেবে ব্যবহৃত হয়। ধনাত্মক ডাটার জন্য সাইন বিটটি শূন্য (০) হয় এবং ঋণাত্মক ডাটার জন্য সাইন বিটটি এক (১) হয়।

উদাহরণ ঃ

```
int Value1= 65;
```

স্টেটমেন্টের মাধ্যমে Value1 নামের কোন int টাইপের ভেরিয়েবল ঘোষণা করলে কম্পাইলার প্রোগ্রাম নির্বাহের সময় মেমোরিতে Value1 নামে ৪ বাইট জায়গা বরাদ্দ করে সেখানে ৬৫ সংরক্ষণ করে।

লং টাইপ (long) ঃ

জাভা ভাষায় খুব বড় পূর্ণসংখ্যা যথা ঃ -৯২২৩৩৭২০৩৬৮৫৪৭৭৫৮০৮ থেকে ৯২২৩৩৭২০৩৬৮৫৪৭৭৫৮০৭ এর মধ্যবর্তী যে কোন পূর্ণসংখ্যা নিয়ে কাজ করার জন্য long টাইপ ভেরিয়েবল ব্যবহৃত হয়। long টাইপের ব্যাপ্তি int টাইপ অপেক্ষা অনেক বেশি। long টাইপ ভেরিয়েবল ঘোষণার জন্য long কীওয়ার্ড ব্যবহৃত হয়। প্রতিটি long টাইপ ভেরিয়েবলের জন্য কম্পাইলার মেমোরিতে ৮ বাইট বা ৬৪ বিট জায়গা সংরক্ষণ করে। ৬৪ বিট জায়গার মধ্যে ডান দিক থেকে প্রথম ৬৩ বিট (বিট ০ থেকে বিট ৬২) ডাটা রাখার জন্য এবং বামের বিটটি (বিট ৬৩) সাইন বিট হিসেবে ব্যবহৃত হয়। ধনাত্মক ডাটার জন্য সাইন বিটটি শূন্য (০) হয় এবং ঋণাত্মক ডাটার জন্য সাইন বিট টি এক (১) হয়।

উদাহরণ ঃ

```
long Value1= 65;
```

স্টেটমেন্টের মাধ্যমে Value1 নামের কোন long টাইপের ভেরিয়েবল ঘোষণা করলে কম্পাইলার প্রোগ্রাম নির্বাহের সময় মেমোরিতে Value1 নামে ৮ বাইট জায়গা বরাদ্দ করে সেখানে ৬৫ সংরক্ষণ করে।

ফ্লোটিং পয়েন্ট টাইপ (float) ঃ

জাভা ভাষায় ৩.৪ X ১০$^{-৩৮}$ থেকে ৩.৪ X ১০$^{+৩৮}$ এর মধ্যবর্তী যে কোন পূর্ণ বা দশমিক মানের ডাটা (যেমন ২৩, ২৩৪৫, ৪৫.২৩ ইত্যাদি) নিয়ে কাজ করার জন্য float টাইপ ভেরিয়েবল ব্যবহৃত হয়। float টাইপ ভেরয়েবল ঘোষণা জন্য float কীওয়ার্ড ব্যবহৃত হয়। প্রতিটি float টাইপ ভেরিয়েবলের জন্য কম্পাইলার মেমোরিতে ৪ বাইট বা ৩২ বিট জায়গা সংরক্ষণ করে। এই ৩২ বিট জায়গার মধ্যে ডান দিক থেকে প্রথম ২৩টি ম্যানটিসা বিট (বিট ০ থেকে বিট ২২) ডাটা রাখার জন্য, পরবর্তী ৮ টি এক্সপোনেনশিয়াল বিট (বিট ২৩ থেকে বিট ৩০) দশমিক বিন্দুর পরের অংশের ডাটার জন্য এবং সবচেয়ে বামের বিটটি (বিট ৩১) সাইন বিট হিসেবে ব্যবহৃত হয়। ধনাত্মক ডাটার জন্য সাইন বিটটি শূন্য (০) হয় এবং ঋণাত্মক ডাটার জন্য সাইন বিট টি এক (১) হয়।

উদাহরণ ঃ

```
float Value1;
```

স্টেটমেন্টের মাধ্যমে Value1 নামের কোন float টাইপের ভেরিয়েবল ঘোষণা করলে কম্পাইলার প্রোগ্রাম নির্বাহের সময় মেমোরিতে Value1 নামে ৩২ বিট জায়গা বরাদ্দ করে।

ডাবল টাইপ (double) ঃ

প্রতিটি double টাইপ ভেরিয়েবলের জন্য কম্পাইলার মেমোরিতে ৮ বাইট বা ৬৪ বিট জায়গা সংরক্ষণ করে। float টাইপ অপেক্ষা double টাইপ ভেরিয়েবলের রেঞ্জ বেশি। double টাইপ ভেরিয়েবল ঘোষণার জন্য double কীওয়ার্ড ব্যবহৃত হয়। জাভা ভাষায় ১.৭ X ১০$^{-৩০৮}$ থেকে ১.৭ X ১০$^{+৩০৮}$ এর মধ্যবর্তী যে কোন পূর্ণ বা দশমিক কিংবা এক্সপোনেনশিয়াল মানের ডাটা (যেমন ২৩, -২৩৪৫, ৪৫.২৩, ১.৫ X ১০^{১০} ইত্যাদি) নিয়ে কাজ করার জন্য double টাইপ ভেরিয়েবল ব্যবহৃত হয়। এই ৬৪ বিট জায়গার মধ্যে ডান দিক থেকে প্রথম ৫২ টি ম্যানটিসা বিট (বিট ০ থেকে বিট ৫১) ডাটা রাখার জন্য, পরবর্তী ১১ টি এক্সপোনেনশিয়াল বিট (বিট ৫২ থেকে বিট ৬২) দশমিক বিন্দুর পরের অংশের ডাটার জন্য এবং সবচেয়ে বামের বিটটি (বিট ৬৩) সাইন বিট হিসেবে ব্যবহৃত হয়। ধনাত্মক ডাটার জন্য সাইন বিটটি শূন্য (০) হয় এবং ঋণাত্মক ডাটার জন্য সাইন বিট টি এক (১) হয়।

উদাহরণ ঃ

```
double Value1;
```

স্টেটমেন্টের মাধ্যমে Value1 নামের কোন double টাইপ ভেরিয়েবল ঘোষণা করলে কম্পাইলার প্রোগ্রাম নির্বাহের সময় মেমোরিতে Value1 নামে ৬৪ বিট জায়গা বরাদ্দ করে।

৬.২ ভেরিয়েবল ডিক্লারেশন ও ডাইনামিক ইনিশিয়ালাইজেশন (Declaration and Dynamic Initialization of Variables in Java) ঃ

ভেরিয়েবল (Variable) ঃ একজন প্রোগ্রামার কাজ করার সুবিধার জন্য মেমোরিতে ডাটা সংরক্ষণের জন্য cell এর অ্যাড্রেস ব্যবহার না করে cell এর একটা নাম দেন এবং ঐ নামের অধীনে ডাটা সংরক্ষণ করেন। সুতরাং ভেরিয়েবল হচ্ছে একটি Container এর ন্যায় যাতে মেমোরি অ্যাড্রেসের পরিবর্তে কোন নামের অধীনে পরিবর্তনশীল ডাটা সংরক্ষণ করা যায়।

ডাটা টাইপের উপর ভিত্তি করে জাভা ভাষায় ভেরিয়েবলকে ৮ ভাগে ভাগ করা যায়। যথা ঃ

- ইন্টিজার (int)
- ক্যারেক্টার (char)
- ফ্লোটিং পয়েন্ট (float)
- ডাবল (double)
- শর্ট (short)
- লং (long)
- বাইট (byte) ও
- বুলিয়ান (boolean)

উদাহরণ ঃ

```
int length, width;
float PI, Area;
double d;
boolean b; ইত্যাদি।
```

ভেরিয়েবল ডিক্লারেশন (Variable declaration) ঃ ডাটা টাইপসহ ভেরিয়েবলের নামকরণ করার প্রক্রিয়াকে ভেরিয়েবল ডিক্লারেশন বলে। প্রোগ্রামে কোন ভেরিয়েবল ডিক্লেয়ার করা হলে সেই ভেরিয়েবলের নামে মেমোরি স্পেস বরাদ্দ হয়। অতঃপর সেই ভেরিয়েবলের মান নির্ধারণ করলে সেই মান বরাদ্দকৃত মেমোরি স্পেসে অবস্থান করে। পরবর্তীতে ঐ ভেরিয়েবলের নামে সেই মানকে প্রোগ্রামে ব্যবহার করা হয়। ভেরিয়েবল ডিক্লারেশনের ফরম্যাট হল ঃ

```
DataType VariableName (s);
```

এখানে 'DataType' বলতে প্রিমিটিভ বা বিল্ট-ইন ডাটা টাইপ (int, char, short, long, boolean) কে বুঝানো হয়। 'VariableName (s)' হচ্ছে এক বা একাধিক ভেরিয়েবলের নাম যা প্রোগ্রামে ব্যবহৃত হবে। একাধিক ভেরিয়েবল হলে কমা (,) ব্যবহার করে লিখতে হবে।

উদাহরণ ঃ

```
int radius;
char 'A';
float r, circletrea; ইত্যাদি।
```

ডাইনামিক ইনিশিয়ালাইজেশন (Dynamic Insitialization) ঃ যখন কোন ভেরিয়েবলকে ডিক্লেয়ার করার সময় একই সাথে এর মান ও নির্ধারণ করে দেয়া হয় তখন তাকে ভেরিয়েবলের ডাইনামিক ইনিশিয়ালাইজেশন (Dynamic Initialization of variable) বলে।

ভেরিয়েবলের ডাইনামিক ইনিশিয়ালাইজেশনের ফরম্যাট হল ঃ

```
Datatype VariableName = Value;
```

উদাহরণ ঃ

```
int length = 10;
float PI = 3.1416;
```

ভেরিয়েবল নামকরণের নিয়মাবলি (Naming convention of variable) ঃ জাভা ভাষায় কোন ভেরিয়েবলকে ক্লাস কিংবা মেথডের মধ্যে সুবিধাজনক অবস্থানে ঘোষণা করা হয় এবং পরে প্রয়োজন অনুযায়ী ব্যবহার করা হয়। ভেরিয়েবল ঘোষণা , নামকরণ এবং তা ব্যবহারের জন্য কতগুলো সুনির্দিষ্ট নিয়ম অনুসরণ করতে হয়। নিম্নে সেগুলো উল্লেখ করা হল ঃ

- ভেরিয়েবল নামকরণে কেবল আলফাবেটিক ক্যারেক্টার (a,.....,z, A,.....Z), ডিজিট (0,1,2,....,9) এবং আন্ডারস্কোর (_) ও ডলার চিহ্ন ($) ব্যবহার করা যায়। আন্ডারস্কোর ও ডলার চিহ্ন ব্যতীত অন্য কোন স্পেশাল ক্যারেক্টার (যেমন, !,@,#,%,*,+,- ইত্যাদি) ব্যবহার করা যায় না। যেমন– my_var, My$Roll বৈধ ভেরিয়েবল; কিন্তু my@var, My&Roll অবৈধ।
- ভেরিয়েবল নামের মধ্যে ফাঁকা স্থান থাকবে না। যেমন, RollNo, Roll_ No ইত্যাদি বৈধ ভেরিয়েবল। কিন্তু Roll 1, Roll No অবৈধ।
- ভেরিয়েবলের নাম অংক বা ডিজিট দিয়ে শুরু হয় না। যেমন– Roll_1 ও Roll_10 বৈধ ভেরিয়েবল; কিন্তু 5Roll ও 3_Roll অবৈধ।
- জাভা প্রোগ্রামে বড় হাতের এবং ছোট হাতের অক্ষরগুলো আলাদা অর্থ বহন করে। তাই Roll_1, roll_10 ও MyRoll এবং roll_1, Roll_10 ও MyRoll ইত্যাদি আলাদা আলাদা ভেরিয়েবল হিসেবে চিহ্নিত হবে।
- কীওয়ার্ডকে ভেরিয়েবল নাম হিসেবে ব্যবহার করা যায় না। এবং main কোন কীওয়ার্ড না হলেও ভেরিয়েবল নাম হিসেবে main ব্যবহার করা যায় না। অবশ্য কীওয়ার্ডসমূহের নামের এক বা একাধিক বর্ণ বড় হরফে লিখে আইডেন্টিফায়ার নাম হিসেবে ব্যবহার করা যায়। তবে এরূপ না করাই উত্তম। Int, Char, Main, MAIN বৈধ ভেরিয়েবল; কিন্তু int, main, private ইত্যাদি অবৈধ।
- ভেরিয়েবল নামকরণে ২৫৫ ক্যারেক্টার পর্যন্ত ব্যবহার করা যায়। তবে এক ধরনের দুইটি ভেরিয়েবল নামের পার্থক্য অবশ্যই প্রথম ৩১ টি ক্যারেক্টারের মধ্যে হতে হয়। এজন্য ভেরিয়েবল নাম করণে ৩১টি ক্যারেক্টারের বেশি ব্যবহার না করাই ভাল।

৬.৩ ভেরিয়েবলের স্কোপ ও লাইফটাইম (Scope and Lifetime of Variable in Java) ঃ

স্কোপ (Scope) ঃ কোন ভেরিয়েবলকে প্রোগ্রামের কোথায় ব্যবহার করা যাবে কিংবা কোথায় ব্যবহার করা যাবে না তা নির্ধারণ করে দেয়ার প্রক্রিয়াই হচ্ছে ঐ ভেরিয়েবলের স্কোপ (scope)। জাভা ভাষায় কোন ভেরিয়েবলের স্কোপ নির্ধারণের জন্য বিভিন্ন ধরনের টাইপ মডিফায়ার (Type Modifier) ব্যবহৃত হয়। যেমন–

- abstract
- final
- native
- static
- synchronized
- transient এবং
- volatile ইত্যাদি।

লাইফ টাইম (Lifetime) ঃ কোন ভেরিয়েবল তার মধ্যে রক্ষিত ভ্যালু (Value) কে যত সময় সংরক্ষণ করতে পারে তাই হল ঐ ভেরিয়েবলের লাইফটাইম (life time)।

কোন ভেরিয়েবলের লাইফ টাইম নির্ভর করে ভেরিয়েবলটি কোথায় ডিক্লেয়ার করা হয়েছে তার উপর। অর্থাৎ যদি ভেরিয়বলটি অ্যাট্রিবিউট হিসেবে ডিক্লেয়ার করা হয় তবে পুরো ক্লাসে ভেরিয়েবলটি একসেস করা যাবে। আর মেথডের ভিতর ডিক্লেয়ার করলে মেথড শেষ হওয়ার সাথে সাথে ভেরিয়বলের লাইফটাইম শেষ হয়ে যাবে।

জাভায় দু ধরনের লাইফ টাইম (Life time) বিশিষ্ট ভেরিয়েবল ব্যবহার করা হয়। যথা ঃ

- যখন instance সৃষ্টি হয় তখন ক্লাসে এক ধরনের ভেরিয়েবল ব্যবহার করা হয় যা ক্লাসের যেকোন মেথড এ চেনা যায় বা নিজের স্বকীয়তা প্রকাশ করে, তাকে মেম্বার ভেরিয়েবল বা ক্লাস ভেরিয়েবল বলে।
- আর এক ধরনের ভেরিয়েবল আছে যা মেথডের মধ্যেই সৃষ্টি হয় এবং মেথডের মধ্যে শেষ হয়। মেথডের বাইরে একে চেনা যায় না, একে বলা হয় অটোমেটিক ভেরিয়েবল বা মেথড ভেরিয়েবল।

৬.৪ জাভায় ব্যবহৃত অপারেটরসমূহ (Operators in Java) ঃ

অপারেটর ও অপারেন্ড (Operator and operand) ঃ

জাভা ভাষায় গাণিতিক এবং যৌক্তিক কাজ নিয়ন্ত্রণ করার জন্য কতগুলো বিশেষ ক্যারেক্টার (যেমন ঃ + , - , *, / , ++, --, >> , >=, <<, <= ইত্যাদি) ব্যবহৃত হয়। এসব বিশেষ ক্যারেক্টারকে বলা হয় অপারেটর। অপারেন্ড বা ডাটা ব্যবহার করে বিভিন্ন কর্ম সম্পাদনের জন্য অপারেটর ব্যবহৃত হয়।

সুতরাং অপারেটর হলো কোন বিশেষ চিহ্ন (symbol) বা শব্দ (word) যা কম্পাইলারকে কোন বিশেষ গাণিতিক বা যৌক্তিক কাজ করতে নির্দেশ করে। যেমনঃ a + b তে + হলো একটি অপারেটর। অপারেটর যেসব ডাটা, ভেরিয়েবল বা expression এর উপর কাজ করে তাদেরকে operand বলে। যেমন ঃ a + b এখানে a ও b হলো operand.

প্রকারভেদ (Classification of operator) ঃ জাভায় এক সেট শক্তিশালী অপারেটর আছে যাদের অধিকাংশই সরাসরি সি এবং সি++ থেকে নেয়া। জাভা অপারেটরগুলোকে তাদের প্রকৃতি ও কার্য অনুযায়ী ৭ ভাগে ভাগ করা যায়। যথা ঃ

- এরিথমেটিক অপারেটর (Arithmetic operator)
- রিলেশনাল অপারেটর (Relational operator)
- লজিক্যাল অপারেটর (Logical operator)
- বিটওয়াইজ অপারেটর (Bitwise operator)
- এসাইনমেন্ট অপারেটর (Assignment operator)
- কন্ডিশনাল অপারেটর (Conditional operator) ও
- স্পেশাল অপারেটর (Special operator) ইত্যাদি।

এরিথমেটিক অপারেটর (Arithmetic operator) ঃ

যেসব অপারেটর বিভিন্ন ধরনের গাণিতিক কাজ (যেমন ঃ যোগ, বিয়োগ, গুণ, ভাগ, ভাগশেষ ইত্যাদি) করার জন্য ব্যবহৃত হয় তাদেরকে গাণিতিক অপারেটর (Arithmetic operator) বলে। গাণিতিক অপারেটর দুই প্রকার। যথা ঃ

- বাইনারী অপারেটর (Binary Operator)
- ইউনারী অপারেটর (Unary Operator)

বাইনারী অপারেটর (Binary Operator) ঃ যেসব অপারেটর একসাথে দুটি অপারেন্ড নিয়ে কাজ করতে পারে তাদের বাইনারি অপারেটর বলে।

[illegible]	[illegible]	[illegible]
Addition	+	দুটি Operand-কে যোগ করে যোগফল দেখায়
Subtraction	–	প্রথম Operand থেকে ২য় Operand বিয়োগ করে বিয়োগফল দেখায়
Multiplication	*	দুটো Operand-কে গুণ করে গুণফল দেখায়
Division	/	প্রথম Operand-কে ২য় Operand দিয়ে ভাগ করে ভাগফল দেখায়
Modulus	%	প্রথম Operand-কে ২য় Operand দিয়ে ভাগ করার পর যা ভাগশেষ থাকে, তা দেখায়।

টেবিল ঃ ৩.১ Binary Operator-এর List

উদাহরণ ঃ

```
public class arithmetic
{
public static void main(String args[])
{
int a=10, b=15, c=20;
float d;
int result;
d=(float)b/(float)a;
System.out.println("a = " +a);
```

```
System.out.println("b = " +b);
System.out.println("c = " +c);
System.out.println("Division: b/a = " +d);
result=b%a;
System.out.println("\n Modulo: b%a = "+ result);
result=a*b+c;
System.out.println("\n Expression : a*b+c =" + result);
result=a+b*c;
System.out.println("\n Expression : a+b*c =" +result);
}
}
```

ইউনারী অপারেটর (Unary Operator) ঃ যেসব অপারেটর একটি মাত্র অপারেন্ড নিয়ে কাজ করে তাদেরকে ইউনারী অপারেটর (Unary operator) বলে।

Operator (অপারেটর)	Symbol (চিহ্ন)	Operation (কাজ)
Increment	++	Operand এর মানের সাথে 1 যোগ করে
Decrement	– –	Operand এর মানের সাথে 1 বিয়োগ করে।
Unary plus	+	Operend এর পূর্বে বসে ধনাত্মক মান প্রকাশ করে।
Unary Minus	–	Operend এর পূর্বে বসে ঋণাত্মক মান প্রকাশ করে।
Bitwise Inversion	~	সকল বাইনারী 0 কে 1 এবং সকল 1 কে 0 এ পরিবর্তন করে।
Boolean complement operator	!	বুলিয়ান মানকে বিপরীত মানে পরিবর্তন করে।

ইনক্রিমেন্ট অপারেটর (Increment Operator) ঃ ইনক্রিমেন্ট অপারেটর (++) হলো এক প্রকার ইউনারি অপারেটর, যা কোন অপারেন্ডের মান শর্তসাপেক্ষে বা শর্তহীনভাবে 1 করে বাড়ায়।

ইনক্রিমেন্ট অপারেটর দুই ধরনের। যথা ঃ

- প্রিফিক্স ইনক্রিমেন্ট (Prefix Increment) ও
- পোস্টফিক্স ইনক্রিমেন্ট (Postfix Increment)।

প্রিফিক্স ইনক্রিমেন্ট অপারেটর (Prefix increment Operator) ঃ যদি ইনক্রিমেন্ট অপারেটরটি (++) অপারেন্ডের পূর্বে বসে তবে তাকে প্রিফিক্স ইনক্রিমেন্ট অপারেটর বলে। যেমন– ++x, ++a, ++y ইত্যাদি।

পোস্টফিক্স ইনক্রিমেন্ট অপারেটর (Postfix increment Operator) ঃ যদি ইনক্রিমেন্ট অপারেটরটি (++) অপারেন্ডের পরে বসে তবে তাকে পোস্টফিক্স ইনক্রিমেন্ট অপারেটর বলে। যেমন– x ++, a++, y ++ ইত্যাদি।

প্রিফিক্স ও পোস্টফিক্স ইনক্রিমেন্টের (++a ও a++ অথবা ++x ও x++) মধ্যে পার্থক্য ঃ প্রিফিক্স ও পোস্টফিক্স ইনক্রিমেন্ট প্রায় একই রকম কাজ করে, তবে এদের মাঝে সামান্য পার্থক্য আছে।

প্রিফিক্স নোটেশনের ক্ষেত্রে কম্পাইলার প্রথমে ভেরিয়েবলের প্রারম্ভিক মানের সাথে 1 যোগ করে অতঃপর প্রোগ্রামের একই স্টেটমেন্ট এই বর্ধিত মান ব্যবহার করে। কিন্তু পোস্টফিক্স নোটেশনের ক্ষেত্রে কম্পাইলার প্রথমে প্রোগ্রামে ভেরিয়েবলের পুরাতন মান ব্যবহার করে, অতঃপর ভেরিয়েবলের মানের সাথে 1 যোগ করে। এই নতুন মান পরবর্তী ধাপে কার্যকর হয়।

এখানে উভয় Expession ই একই কাজ করে।

ডিক্রিমেন্ট অপারেটর (Decrement operator) ঃ ডিক্রিমেন্ট অপারেটর (- -) হল এক প্রকার ইউনারি অপারেটর, যা কোন অপারেন্ডের মান শর্তসাপেক্ষে বা শর্তহীনভাবে 1 করে কমায়।

ডিক্রিমেন্ট অপারেটর দুই প্রকার। যথা ঃ

- প্রিফিক্স ডিক্রিমেন্ট (Prefix Decrement) ও
- পোস্টফিক্স ডিক্রিমেন্ট (Postfix Decrement)।

প্রিফিক্স ডিক্রিমেন্ট অপারেটর (Prefix Decrement Operator) ঃ যদি ডিক্রিমেন্ট অপারেটরটি (- -) অপারেন্ডের পূর্বে বসে তবে তাকে প্রিফিক্স ডিক্রিমেন্ট অপারেটর বলে। যেমন- - -x, - -a, --y ইত্যাদি।

পোস্টফিক্স ডিক্রিমেন্ট অপারেটর (Postfix Decrement Operator) ঃ যদি ডিক্রিমেন্ট অপারেটরটি (- -) অপারেন্ডের পরে বসে তবে তাকে পোস্টফিক্স ডিক্রিমেন্ট অপারেটর বলে। যেমন- x - -, a- - , y - - ইত্যাদি।

প্রিফিক্স ও পোস্টফিক্স ডিক্রিমেন্টের (- - a ও a- - অথবা - - x ও x- -) মধ্যে পার্থক্য ঃ প্রিফিক্স নোটেশনের ক্ষেত্রে কম্পাইলার প্রথমে ভেরিয়েবলের প্রারম্ভিক মানের সাথে 1 বিয়োগ করে অতঃপর প্রোগ্রামের একই স্টেটমেন্ট এই বর্ধিত মান ব্যবহার করে। কিন্তু পোস্টফিক্স নোটেশনের ক্ষেত্রে কম্পাইলার প্রথমে প্রোগ্রামে ভেরিয়েবলের পুরাতন মান ব্যবহার করে, অতঃপর ভেরিয়েবলের মানের সাথে 1 বিয়োগ করে। এই নতুন মান পরবর্তী ধাপে কার্যকর হয়।

এখানে উভয় Expession ই একই কাজ করে।

উদাহরণ ঃ

```
public class post
{
public static void main(String args[])
{
int counter=0;
System.out.println("Counter is: "+counter);
System.out.println("\n++Counter is: "+ ++counter);
System.out.println("\nCounter is: "+counter);
counter=0;
System.out.println("\ncounter is: "+ counter);
System.out.println("\ncounter++ is:"+counter++);
System.out.println("\ncounter is: "+counter);
}
}
```

রিলেশনাল অপারেটর (Relational Operator) ঃ যেসব অপারেটর দুটো অপারেন্ডের মধ্যে তুলনা বা বিভিন্ন রকম সম্পর্ক (যেমন ঃ ছোট, বড়, ছোট বা সমান, বড় বা সমান ইত্যাদি।) বুঝানোর জন্য ব্যবহৃত হয় তাদেরকে রিলেশনাল অপারেটর বলে। রিলেশনাল অপারেটরের বাম দিকে ও ডানদিকে একটি করে অপারেন্ড থাকে। তবে অপারেন্ডটি কোন ভেরিয়েবল, কন্সট্যান্ট বা কোন এক্সপ্রেশন হতে পারে।

C-তে মোট ৬ টি Relational Operator আছে। যথা ঃ

Operator (অপারেটর)	Symbol (চিহ্ন)	Example (উদাহরণ)	Operation (কাজ)
Equal	= =	a = = b	a এর মান b এর সমান।
Greater than	>	a > b	a এর মান b এর চেয়ে বড়।
Less than	<	a <b	a এর মান b এর চেয়ে ছোট।
Greater than or equal to	> =	a > = b	a এর মান b এর বড়/অথবা সমান।
Less than or equal to	< =	a < = b	a এর মান b এর ছোট অথবা সমান।
Not equal	! =	a! = b	a এর মান b এর সমান নয়।

টেবিল ঃ জাভার Relational Operator

উদাহরণ ঃ মনে করি, P ও Q দুটি int টাইপ ভেরিয়েবল এবং P-এর মান 10 এবং Q-এর মান 5.

Relational Expression	True/False	Result
P == Q	False	0
P! = Q	True	1
P >Q	True	1
P > = Q	True	1
P*15 > Q* 100	False	0

লজিক্যাল অপারেটর (Logical Operator) ঃ যেসব অপারেটর লজিক্যাল অপারেশন (যেমন ঃ লজিক্যাল এন্ড, লজিক্যাল অর, লজিক্যাল নট) সম্পন্ন করার জন্য ব্যবহৃত হয়, তাদের লজিক্যাল অপারেটর বলে।

Logical Operator সাধারণত দুই বা ততোধিক Logical Expression-কে সংযুক্ত করে এবং একটি ফলাফল প্রদান করে।

Operator (অপারেটর)	Symbol (চিহ্ন)	Example (উদাহরণ)
AND	&&	a>b && b>c
OR	||	a>b || b>c
NOT	!	! a

টেবিল ঃ Logical Operator-এর List

লজিক্যাল AND অপারেশন ঃ AND এর ক্ষেত্রে উভয় Expression এর মান 1 হলে ফলাফল 1 হবে।

a	b	(Result) a && b
1	0	0
0	1	0
0	0	0
1	1	1

টেবিল ঃ AND table

লজিক্যাল OR অপারেশন ঃ OR এর ক্ষেত্রে কমপক্ষে যে কোন একটি expression এর মান 1 হলে ফলাফল 1 হবে।

a	b	(Result) a || b
1	0	1
0	1	1
0	0	0
1	1	1

টেবিল ঃ OR table

লজিক্যাল NOT অপারেশন ঃ NOT সত্যকে মিথ্যা কিংবা মিথ্যাকে সত্যে পরিণত করে।

Input		Result	
a	b	! a	! b
0	1	1	0
1	0	0	1

টেবিল ঃ NOT table

উদাহরণ ঃ মনে করি, p = 5, q = 6, r = 4. তাহলে experession-গুলো নিম্নরূপ ফলাফল প্রদর্শন করবে।

Expression	True/False	Result
(p>q) || (p<q)	True	1
(p>q) && (p<q)	False	0
(p>r) || (p + q) > = 5)	True	1
!p	False	0

টেবিল ঃ Logical Expression-এর উদাহরণ

বিটওয়াইজ অপারেটর (Bitwise Operator) ঃ বাইনারি ডাটা অর্থাৎ বিট বা বাইট নিয়ে বিভিন্ন রকমের লজিক্যাল অপারেশন (যেমন– এন্ড, অর, নট, এক্স নর, লেফট শিফট, রাইট শিফট ইত্যাদি) সম্পন্ন করার জন্য যেসব অপারেটর ব্যবহার করা হয়, তাদেরকে বিটওয়াইজ অপারেটর (Bitwise operator) বলে।

Operator (অপারেটর)	Symbol (চিহ্ন)
&	Bitwise AND
|	Bitwise OR
^	Bitwise XOR
<<	Left shift
>>	Right shift
~	Inverse

এখানে Inverse (~) হল Unary operator এবং অন্য Operator গুলো হল Binary operator; Bitwise operator গুলো শুধুমাত্র int টাইপ এর Operand নিয়ে কাজ করে। নিম্নে বিভিন্ন Bitwise operator নিয়ে আলোচনা করা হল ঃ

Bitwise AND (&) (অপারেশন) ঃ

মূলত দুটো বিটকে গুণ করার জন্য Bitwise AND ব্যবহার করা হয়। Bitwise AND ব্যবহার করে গঠিত expression এর মান নিচের তালিকা অনুযায়ী পাওয়া যায়।

Input		Output
a	b	a&b
0	0	0
0	1	0
1	0	0
1	1	1

অর্থাৎ a এবং b এর মান 0 হলে a & b এর মান হবে 0 একইভাবে অন্য মানগুলো কাজ করে। তবে উপরের এই তালিকা দেখতে Logical AND (&&) operator এর মত হলেও, Logical AND (&&) এবং Bitwise AND (&) এর মধ্যে কিছুটা ভিন্নতা আছে। যেমন-

x = 1, y = 2 হলে,

x&&y এর মান হবে 1; কিন্তু

x&y এর মান হবে 0.

কারণ, Logical AND (&&) তার Operand এর মানকে ডেসিমাল সংখ্যা হিসাবে নেয়। এক্ষেত্রে 0 হল 0 বা মিথ্যা) এবং অন্য যে কোন পূর্ণ সংখ্যা যেমন- 1, 2, 3, 4 .. .. .. ইত্যাদি হল 1 (বা সত্য)। কিন্তু Bitwise AND (&) তার Operand এর মানকে বাইনারি সংখ্যা হিসাবে নেয়। এক্ষেত্রে সংখ্যাটিকে বাইনারিতে পরিণত করে সংখ্যার প্রতিটি বিটকে গুণ করা হয় এবং Expression ফলাফল বের করা হয়। যেমন-

x = 1, y = 2 হলে,

x = 1 ⇒ 0000 0001
y = 2 ⇒ 0000 0010
x & y ⇒ 0000 0000 [1&0 = 0 and 0 & 1 = 0]

আরেকটি উদাহরণ ঃ

a = 5, b = 7 হলে, a&b = 5

a = 5 ⇒ 0000 0101
b = 7 ⇒ 0000 0111
a & b ⇒ 0000 0101

Bitwise OR (|) (অপারেশন) ঃ প্রোগ্রামে দুটো বিটকে যোগ করার জন্য Bitwise OR (|) ব্যবহার করা হয়। AND এর মত Bitwise OR (|) ও দুটো operand নিয়ে কাজ করে এবং উভয় Operand ই int টাইপ হতে হয়। Bitwise OR (|) দ্বারা গঠিত Expression এর মান নিচের তালিকানুযায়ী পাওয়া যায়।

Input		Output
a	b	a | b
0	0	0
0	1	1
1	0	1
1	1	1

উপরের এই তালিকা Logical OR (||) এর মত হলেও, উভয় Operator দ্বারা গঠিত Expression ভিন্ন ভিন্ন মান প্রদান করে। যেমন-

a = 3, b = 4 হলে,

a || b = 1 কিন্তু

a | b = 7

Logical OR (||) এর মান হয়ত 0 (মিথ্যা) অথবা 1 (সত্য) হয়। কিন্তু Bitwise OR (|) এর ক্ষেত্রে Operand এর মানকে প্রথমে বাইনারিতে পরিণত করা হয় এবং পরে প্রতিটি বিটকে যোগ করে Expression এর ফলাফল বের করা হয়। যেমন-

a = 4 ⇒ 0000 0100 (4)

b = 3 ⇒ 0000 0011(3)

a | b ⇒ 0000 0111(7)

Bitwise XOR (∧) (অপারেশন) ঃ

Bitwise XOR (∧) যাকে Exclusive OR- ও বলা হয়। এতে Expression এর মান নিচের তালিকানুযায়ী পাওয়া যায়।

Input		Output
a	b	a ∧ b
0	0	0
0	1	1
1	0	1
1	1	0

অর্থাৎ XOR (∧) Operator-এর উভয় Operand bit যদি একই হয়, তাহলে Expression এর মান হবে 0 অন্যথায় 1 হবে। যেমন-

a = 5, b = 7 হলে, a ∧ b = 2 হবে

a = 5 ⇒ 0000 0101 (5)

b = 7 ⇒ 0000 0111 (7)

a ∧ b ⇒ 0000 0010 (2)

Exclusive OR কে মূলত বিভিন্ন প্রোগ্রামে Encrypted পদ্ধতির Password তৈরিতে ব্যবহার করা হয়। কেননা এই পদ্ধতিতে Password encode করলে তা সহজে Decode করা যায় না। এক্ষেত্রে Password encode করার জন্য একটি Value ব্যবহার করা হয় যাকে Mask value বলে এবং এই Mask value ই Encryption key হিসাবে ব্যবহার করা হয়।

Bitwise Inverse (~) (অপারেশন) ঃ Bitwise Inverse (~) হল Unary operator অর্থাৎ এটা একটা Operand নিয়ে কাজ করে। এই Operatorটি তার operand এর বিট 0 থাকলে 1 এবং 1 থাকলে তা 0 তে পরিবর্তন করে দেয়। যেমন-

a = 15 হলে, ~a = 240 হবে

a = 15 ⇒ 0000 111 (15)

~ a = 240 ⇒ 1111 000 (240)

একইভাবে,

a = 254 হলে, ~ a = 1 হবে,

a = 254 ⇒ 11 1110 (254)

~ a = 1 ⇒ 00 0001 (1)

Bitwise Shift (<< এবং >>) (অপারেশন) ঃ প্রোগ্রামে অনেক সময় এক বা একাধিক বিট ডানদিকে বা বামদিকে Shift করতে হয়। এ কাজে Shift (<<, >>) Operator দুটো ব্যবহার করা যায়। Left shift (<<) Operator ডাটা বিটকে বামদিকে এবং Right shift (>>) Operator ডাটা বিটকে ডানদিকে Shift করে। যেমন-

a << 2 মানে হল, a এর বিটগুলো দুই ঘর বামদিকে সরে আসবে এবং খালি জায়গায় 0 বসবে।

a << 5 মানে হল, a এর বিটগুলো পাঁচ ঘর বামদিকে সরে আসবে এবং খালি জায়গায় 0 বসবে।

এবং

a >> 2 মানে হল a এর বিটগুলো দুই ঘর ডানদিকে সরে আসবে এবং খালি জায়গায় 0 বসবে।

a >> 5 মানে হল, a এর বিটগুলো পাঁচ ঘর ডানদিকে সরে আসবে এবং খালি জায়গায় 0 বসবে।

উদাহরণ ঃ

a = 2 হলে, a << 2 = 8 হবে,

a = 2 ⇒ 0000 0010 (2)

a << 2 ⇒ 0000 1000 (8)

এখানে বামদিকে দুই বিট Shift করা হয়েছে।

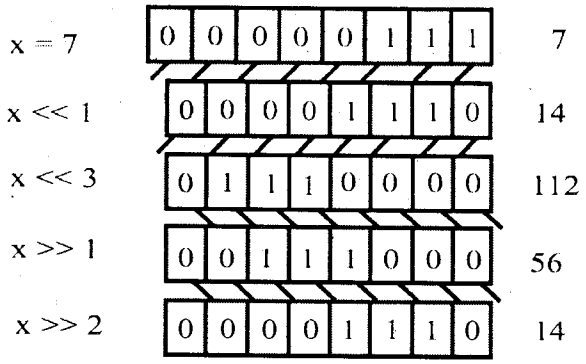
একইভাবে a = 4 হলে, a >> 2 = 1 হবে,

a = 4 ⇒ 0000 100 (4)

a >> 1 ⇒ 0000 001 (1)

এখানে ডানদিকে দুই বিট Shift করা হয়েছে।

Shift operator গুলো আরও ভালোভাবে বোঝার জন্য নিচের উদাহরণটি দেখা যেতে পারে ঃ



চিত্র ঃ ৬.৩

এভাবে Shift operator ব্যবহার করে বিট Shifting এর কাজ করা হয়।

এসাইনমেন্ট অপারেটর (Assignment Operator) ঃ

কোন Expression বা ভেরিয়েবল এর মানকে অপর কোন ভেরিয়েবল এর মান হিসাবে নির্ধারণ করার জন্য যে ধরনের Operator '=' ব্যবহার করা হয় তাকেই অ্যাসাইনমেন্ট অপারেটর বলে।

জাভায় Assignment Operator হিসেবে '=' কে ব্যবহার করা হয়।

উদাহরণ ঃ

```
int x = 5, y = 2, z;
z = x + y ;
```

এখানে, x এর মান 5 ও y এর মান রাখা হয়েছে 2

প্রথমে, x ও y এর মান যোগ হবে এবং পরে এই সম্মিলিত মান z-এর মধ্যে রাখা হবে। এখানে, z এর মান হবে 7.

Assignment Operator লেখার সাধারণ নিয়ম হল ঃ

```
Variable = Expression;
```

শর্টহ্যান্ড অ্যাসাইনমেন্ট অপারেটর (Short hand Assignment Operator) ঃ জাভা প্রোগ্রামে অ্যাসাইনমেন্ট অপারেটরের সাথে গাণিতিক অপারেটর ব্যবহার করার রীতিকে শর্টহ্যান্ড অ্যাসাইনমেন্ট অপারেটর বলা হয়। যেমন ঃ x = x + 1 কে শর্টহ্যান্ড অপারেটরের সাহায্যে x + = 1 আকারে লেখা হয়। নিচে শর্টহ্যান্ড অপারেটরের তালিকা দেয়া হল ঃ

অপারেটর	অ্যাসাইনমেন্ট অপারেটর	শর্টহ্যান্ড অ্যাসাইনমেন্ট অপারেটর
Addition and assignment	y = y + x	y + = x
Subtraction and assignment	y = y – x	y – = x
Multiplication and assignment	y = y*x	y* = x
Division and assignment	y = y/x	y/ = x
Modulus and assignment	y = y%x	y% = x

কন্ডিশনাল অপারেটর (Conditional Operator) ঃ বিভিন্ন ধরনের শর্তযুক্ত কাজ করার জন্য জাভায় যে ধরনের Operator ব্যবহার করা হয় তাকে Conditional operator বলে।

জাভায় Conditional operator হল (? ঃ)। Conditional operator একই সাথে তিনটি Operand নিয়ে কাজ করতে পারে বলে একে Ternary operator ও বলা হয়। এর গঠন নিম্নরূপ ঃ

Expression 1? Expression 2 : Expression 3;

এখানে, প্রথমে Expression 1 যদি True (non zeros) হয় তবে Expression 2 মূল্যায়িত হবে এবং সম্পূর্ণ Conditional expression -এর মান হবে Expression 2 এর মান। কিন্তু Expression 1 যদি False (zero) হয় তবে expression 3 মূল্যায়িত হবে এবং সম্পূর্ণ Conditional expression -এর মান হবে expression 3 -এর মান।

মনে করি, a = 5, b = 6 ;

```
c = (a > b) ? a : b ;
```

এখানে, a > b কথাটি False,. তাই b এর মানই হবে c এর মান অর্থাৎ c এর মান 6.

উদাহরণ ঃ

x এর মান	y এর মান	Statement	Result
x = 10	y = 5	z = (x > y)? x : y	z = 10
x = 5		z = (x < 0)? 0 : 1	z = 1
	y = 1	z = y? 1 : 5	z = 1

টেবিল ঃ ৩.১১ Conditional Operator-এর উদাহরণ

স্পেশাল অপারেটর (Special operator) ঃ জাভায় ব্যবহৃত স্পেশাল অপারেটরগুলো হল ঃ

- কমা অপারেটর (Comma operator)
- ইন্সট্যান্স অফ অপারেটর (instance of operator)
- ডট অপারেটর (Dot (.) operator)

কমা অপারেটর (Comma operator) ঃ

একই ডাটা টাইপের বিভিন্ন ভেরিয়েবলকে পৃথক করার জন্য কমা অপারেটর (,) ব্যবহৃত হয়। আবার অনেক সময় বিভিন্ন এক্সপ্রেশনের মধ্যে সমন্বয় সাধন করতেও কমা অপারেটর ব্যবহৃত হয়।

এক্সপ্রেশন (Expression)	কমা অপারেটর ব্যবহার করে [illegible]
int c = 8; int a = 3; int b = 5;	int c = 8, a = 3, b = 5;
int a = 15; int b = 8; int sum = a + b;	int a = 15, b = 8, sum = a + b;

Temp = a;
a = b;
b = Temp;
	Temp = a, a = b, b = Temp;

int i; int j = 10; for (i = 1; i < j; i++)	int i, j; for (j = 10, i = 1; i < = j; i ++)

টেবিল ৩.১৬

ইন্সট্যান্স অফ অপারেটর (Instance of Operator) ঃ কোন অবজেক্ট কোন ক্লাসের আওতাভুক্ত কিনা তা নির্ধারণের জন্য instance of অপারেটর ব্যবহৃত হয়। কোন অবজেক্ট কোন ক্লাসের আওতাভুক্ত (instance) হলে instance of অপারেটর 'True' আর না হলে 'False' মান রিটার্ণ করে।

অপারেটরটির বামদিকে থাকবে অবজেক্ট রেফারেন্স, ভেরিয়েবল অথবা অ্যারে আর ডানদিকে থাকবে ক্লাস, ইন্টারফেস অথবা অ্যারে।

ডট অপারেটর (Dot Operator) ঃ কোন ক্লাসের ইন্সট্যান্স ভেরিয়েবল এবং মেথড এক্সেস (Access) করার জন্য (.) ডট অপারেটর ব্যবহৃত হয়।

৬.৫ জাভা ভাষার কন্ট্রোল স্টেটমেন্টসমূহ (Control Flow Statements in Java) ঃ

জাভা প্রোগ্রামের স্টেটমেন্টসমূহ সাধারণত স্বয়ংক্রিয়ভাবে ও পর্যায়ক্রমে একবার করে সম্পাদিত হয়। কিন্তু যদি কোন স্টেটমেন্ট দুই বা ততোধিকবার সম্পাদনের প্রয়োজন হয়, কিংবা কোন স্টেটমেন্ট কোন শর্ত সাপেক্ষে কিংবা অপর কোন স্টেটমেন্টের ফলাফলের ভিত্তিতে সম্পাদনের প্রয়োজন হয়, কিংবা সম্পাদনে বিরত রাখতে হয়, অথবা কোন স্টেটমেন্ট হতে প্রোগ্রামের নিয়ন্ত্রণ, অন্য কোন স্টেটমেন্টে স্থানান্তরের প্রয়োজন হয়, সে সব ক্ষেত্রে স্টেটমেন্টসমূহের নির্বাহ প্রোগ্রামার কর্তৃক নিয়ন্ত্রিত হয়ে হয়। প্রোগ্রামে এরূপ স্টেটমেন্ট নির্বাহ নিয়ন্ত্রণের জন্য কন্ট্রোল স্টেটমেন্ট ব্যবহৃত হয়। স্টেটমেন্ট নিয়ন্ত্রণ যত সহজ হয় প্রোগ্রাম তত সহজ ও সুন্দর হয়। জাভা প্রোগ্রামের কন্ট্রোল স্টেটমেন্টসমূহকে প্রধানত দুইভাগে ভাগ করা যায়। যথাঃ

- কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট (Conditional Control Statement)
- লুপ কন্ট্রোল স্টেটমেন্ট (Loop Control Statement)

কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট (Conditional Control Statement) ঃ

জাভা প্রোগ্রামে শর্তসাপেক্ষে কোন স্টেটমেন্ট সম্পাদনের জন্য কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট ব্যবহৃত হয়। কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট ব্যবহৃত শর্ত সত্য হলে প্রোগ্রামে এক ধরনের ফল পাওয়া যায় আর সত্য না হলে অন্য ধরনের ফল পাওয়া যায়। অন্যতম কন্ডিশনাল কন্ট্রোল স্টেটমেন্টসমূহ হলঃ

- if স্টেটমেন্ট (if statement)
- if-else স্টেটমেন্ট (if-else statement)
- else if স্টেটমেন্ট (else if statement)
- switch স্টেটমেন্ট (switch statement)

if স্টেটমেন্ট (if statement) ঃ জাভা প্রোগ্রামে "যদি" অর্থে if স্টেটমেন্ট ব্যবহৃত হয়। প্রোগ্রামে শর্ত সাপেক্ষে কোন স্টেটমেন্ট সম্পাদনের জন্য if স্টেটমেন্ট ব্যবহৃত হয়।

সিনটেক্স (Syntax) ঃ if স্টেটমেন্ট ব্যবহারের ফরম্যাট হল ঃ

```
if(Condition)
{
Block1;
}
Block2;
```

if স্টেটমেন্ট ব্যবহৃত শর্ত (Condition) সাধারণত এক বা একাধিক লজিক্যাল বা রিলেশনাল এক্সপ্রেশন হয় যা if পরবর্তী প্রথম বন্ধনীর মধ্যে লেখা হয়। if (Condition) স্টেটমেন্টের পর কোন সেমিকোলন থাকবে না। এই শর্তের মান যদি সত্য হয় তবে Block1 সম্পাদিত হয়। অন্যথায় Block2 সম্পাদিত না হয়ে পরবর্তী স্টেটমেন্টসমূহ সম্পাদিত হয়। Block1 যে কোন বৈধ সিম্পল বা কম্পাউন্ড স্টেটমেন্ট হতে পারে।

প্রোগ্রাম ঃ

```
import java.io.*;
public class ifstate
{
public static void main(String args[])
{
int age=0;
String s;
try
{
InputStreamReader in =new InputStreamReader(System.in);
BufferedReader br=new BufferedReader(in);
System.out.print("Enter Your Age");
s=br.readLine();
age=Integer.parseInt(s);
if(age<=0)
System.out.println("You have not borned yet !");
if(age>0 && age<=12)
```

```
System.out.println("You are a child");
if(age>12 && age<=19)
System.out.println("You are a teen ager");
if(age>19 && age<=40)
System.out.println("You are Young");
if(age>40)
System.out.println("Wish Your long life");
}
catch(Exception E){}
}
}
```

if-else স্টেটমেন্ট (if-else statement) ঃ জাভা প্রোগ্রামে "অন্যথায়" অর্থে if স্টেটমেন্টের সাথে else স্টেটমেন্ট ব্যবহৃত হয়। এজন্য এই স্টেটমেন্টকে if-else স্টেটমেন্ট বলা হয়।

সিনটেক্স (Syntax) ঃ if-else স্টেটমেন্ট ব্যবহারের ফরম্যাট হল ঃ

```
if (Condition)
{
Block1;
}
else
{
Block2;
}
Block3;
```

if-else কন্ট্রোল স্টেটমেন্টে ব্যবহৃত শর্ত (Condition) সাধারণত এক বা একাধিক লজিক্যাল বা রিলেশনাল এক্সপ্রেশন হয় যা if এর পরে প্রথম বন্ধনীর মধ্যে লেখা হয়। if (Condition) স্টেটমেন্টের পরে কোন সেমিকোলন বসে না। এই শর্তের মান যদি সত্য হয় তবে Block1 সম্পাদিত হয়। অন্যথা, Block1 সম্পাদিত না হয়ে Block2 সম্পাদিত হয়। উভয় ক্ষেত্রে Block3 সহ পরবর্তী স্টেটমেন্টসমূহ সম্পাদিত হয়। Block1 ও Block2 যে কোন বৈধ সিম্পল বা কম্পাউন্ড স্টেটমেন্ট হতে পারে।

আবার একটি if-else স্টেটমেন্টের মধ্যে অপর একটি if-else স্টেটমেন্টও থাকতে পারে। এরূপ মধ্যবর্তী if-else স্টেটমেন্টকে নেস্টেড if-else স্টেটমেন্ট বলা হয়।

প্রোগ্রাম ঃ if-else স্টেটমেন্ট ব্যবহার করে নিম্নে একটি প্রোগ্রাম দেয়া হল ঃ

```
import java.io.*;
public class ifelse
{
public static void main(String args[])
{
int year=0;
String s;
try
{
```

```
InputStreamReader in =new InputStreamReader(System.in);
BufferedReader br=new BufferedReader(in);
System.out.println("Enter the Year to be tested");
s=br.readLine();
year=Integer.parseInt(s);
if(((year % 4)==0 && (year % 100)!=0) || (year % 400)==0)
System.out.println(""+year+ " is a Leap Year");
else
System.out.println(" +year+ " is not a Leap Year ");
}
catch(Exception E){}
}
}
```

else if স্টেটমেন্ট (else if statement) ঃ জাভা প্রোগ্রামে "অন্যথায় যদি" অর্থে if... else স্টেটমেন্টের সাথে else if স্টেটমেন্ট ব্যবহৃত হয়। else if স্টেটমেন্ট if-else স্টেটমেন্টের if এবং else স্টেটমেন্টের মাঝে বসে। else if স্টেটমেন্ট else এবং if এর মাঝে ফাঁকা স্থান থাকে। প্রোগ্রামে একাধিক if স্টেটমেন্ট ব্যবহারের বিকল্প হিসেবে else if স্টেটমেন্ট জনপ্রিয়। অর্থাৎ প্রোগ্রামে একাধিক শর্ত যাচাই করার জন্য else if ব্যবহৃত হয়।

সিনটেক্স (Syntax) ঃ else if স্টেটমেন্ট ব্যবহারের ফরম্যাট হল ঃ

```
if(Condition1)
{
Block1;
}
else if (Condition2)
{
Block2;
}
else if(Condition3)
{
Block3;
}
.....
else
{
Default Block;
}
BlockN;
.....
```

else if স্টেটমেন্ট ব্যবহৃত শর্তসমূহ সাধারণত এক বা একাধিক লজিক্যাল বা রিলেশনাল এক্সপ্রেশন হয় , যা if এবং else if এর পরে প্রথম বন্ধনীর মধ্যে লেখা হয়। এই শর্তসমূহের মধ্যে যদি কোন শর্ত সত্য হয় তবে সংশ্লিষ্ট স্টেটমেন্ট সম্পাদিত হয় এবং অন্যান্য স্টেটমেন্ট সম্পাদিত হয় না। কিন্তু যদি কোন শর্তই সত্য না হয় তবে else এর পরবর্তী Default Block স্টেটমেন্ট সম্পাদিত হয়। প্রোগ্রামে যে কোন সংখ্যক else if স্টেটমেন্ট থাকতে পারে। তবে if কিংবা else কিংবা if-else সম্পর্কিত একটি মাত্র স্টেটমেন্ট সম্পাদিত হয়। Block1, Block2, Block3 ইত্যাদি কোন বৈধ সিম্পল বা কম্পাউন্ড স্টেটমেন্ট হতে পারে।

প্রোগ্রাম ঃ নিম্নে else if কন্ট্রোল ব্যবহার করে একটি প্রোগ্রাম দেয়া হল ।

```
import java.io.*;
public class elseif
{
public static void main(String args[])
{
int age=0;
String s;
try
{
InputStreamReader in =new InputStreamReader(System.in);
BufferedReader br=new BufferedReader(in);
System.out.println("Enter Your Age");
s=br.readLine();
age=Integer.parseInt(s);
if(age<=0)
System.out.println("You have not borned yet !");
else if(age>0 && age<=12)
System.out.println("You are a child");
else if(age>12 && age<=19)
System.out.println("You are a teen ager");
else if(age>19 && age<=40)
System.out.println("You are Young");
else if(age>40)
System.out.println("Wish your long life");
}
catch(Exception E){}
}
}
```

switch স্টেটমেন্ট (Switch statement) ঃ জাভা প্রোগ্রামে else if স্টেটমেন্টের অনুরূপ কাজে অর্থাৎ একাধিক স্টেটমেন্ট থেকে নির্দিষ্ট কোন স্টেটমেন্ট সম্পাদনের জন্য switch স্টেটমেন্ট ব্যবহৃত হয়। মূলত বেশি সংখ্যক else if স্টেটমেন্ট ব্যবহারের পরিবর্তে switch স্টেটমেন্ট ব্যবহৃত হয়। switch স্টেটমেন্টের সাথে অতিরিক্ত case, break ও default ব্যবহৃত হতে পারে। else if স্টেটমেন্টে কোন কন্ডিশনাল কিংবা রিলেশনাল এক্সপ্রেশনের উপর ভিত্তি করে উপযুক্ত স্টেটমেন্ট নির্বাচন করা হয়। কিন্তু switch স্টেটমেন্টে সাধারণত কোন বৈধ ভেরিয়েবলের মানের ভিত্তিতে উপযুক্ত স্টেটমেন্ট নির্বাচন করা হয়।

সিনটেক্স (Syntax) ঃ switch স্টেটমেন্ট এর ফরম্যাট হল ঃ

```
DataType IndexVariable;
switch(SwitchExp)
{
case1:
{
Block1;
break;
}
case2:
{
Block2;
break;
}
case3:
{
Block3;
break;
}
default:
{
DefaultBlock;
}
}
Block N;
.....
```

এখানে IndexVariable switch স্টেটমেন্ট শুরুর পূর্বে উপযুক্ত ডাটা টাইপসহ ঘোষণা করতে হয়। switch স্টেটমেন্টের সাথে প্রথম বন্ধনীর মধ্যে ব্যবহৃত এক্সপ্রেশনকে সুইচ এক্সপ্রেশন বলা হয় যা ইনডেক্স ভেরিয়েবলের সাথে সম্পর্কিত। case স্টেটমেন্টে ইনডেক্স ভেরিয়েবলের সম্ভাব্য মান দেয়া হয়। case কীওয়ার্ড ও ইনডেক্স ভেরিয়েবলের সম্ভাব্য মানের মাঝে ন্যূনতম একটি স্পেস থাকে এবং কোলন দ্বারা শেষ হয়। case স্টেটমেন্টের শেষে সাধারণত একটি default স্টেটমেন্ট থাকে সেটিও কোলন দ্বারা শেষ হয়। কোন সুইচ স্টেটমেন্টে সর্বাধিক ২৫৬ টি case ব্যবহৃত হতে পারে, তবে default স্টেটমেন্ট থাকবে একটি।

প্রতিটি case স্টেটমেন্টের পর এক বা একাধিক সিম্পল বা কম্পাউন্ড স্টেটমেন্ট থাকে এবং সাধারণত break স্টেটমেন্ট দ্বারা শেষ হয়। তবে এর ব্যবহার অত্যাবশ্যক নয়। ইনডেক্স ভেরিয়েবলের মানের ভিত্তিতে সংশ্লিষ্ট case স্টেটমেন্টের স্টেটমেন্টসমূহ সম্পাদিত হয়। break স্টেটমেন্ট প্রোগ্রাম পয়েন্টার কে switch স্টেটমেন্টের বাইরে বা শেষে স্থানান্তর করে। ফলে পরবর্তী সকল case স্টেটমেন্ট এবং default স্টেটমেন্টসমূহ সম্পাদিত হয় না। ইনডেক্স ভেরিয়েবলের মান যদি case স্টেটমেন্টসমূহের কোনটির সাথে না মিলে তবে default স্টেটমেন্টসমূহ সম্পাদিত হয়। default স্টেটমেন্টের পর break স্টেটমেন্ট অনাবশ্যক। case স্টেটমেন্টে ইনডেক্স ভেরিয়েবলের দুই বা ততোধিক মানের জন্য একই প্রোগ্রাম স্টেটমেন্ট হলে case স্টেটমেন্টসমূহ পরপর লিখে কেবল শেষের case স্টেটমেন্টের জন্য স্টেটমেন্ট উল্লেখ করলেই চলে। একটি case স্টেটমেন্টের মধ্যে অপর কোন switch স্টেটমেন্ট ব্যবহৃত হতে পারে।

প্রোগ্রাম ঃ নিম্নে switch স্টেটমেন্ট ব্যবহার করে একটি প্রোগ্রাম দেয়া হল।

```
import java.io.*;
public class tswitch
{
public static void main(String args[])
{
int year=0;
String s;
try
{
InputStreamReader IN=new InputStreamReader(System.in);
BufferedReader BR=new BufferedReader(IN);
System.out.print("Enter Your Academic Year(From 1 to 4):");
s=BR.readLine();
year=Integer.parseInt(s);
switch(year)
{
case 1:
System.out.println("You are a student of first year");
break;
case 2:
System.out.println("You are a student of Second year");
break;
case 3:
System.out.println("You are a student of Third year");
break;
case 4:
System.out.println("You are a student of Fourth year");
break;
default:
System.out.println("Wrong Input");
}
}
catch(Exception e){}
}
}
```

লুপ কন্ট্রোল স্টেটমেন্ট (Loop Control Statement) ঃ জাভা প্রোগ্রামে কোন স্টেটমেন্ট দুই বা ততোধিক বার সম্পাদনের জন্য লুপ কন্ট্রোল স্টেটমেন্ট ব্যবহৃত হয়। লুপ স্টেটমেন্টসমূহে সাধারণত দুইটি অংশ থাকে। যথা ঃ

- লুপ বডি (Loop Body) এবং
- টেস্ট কন্ডিশন (Test Condition)

প্রোগ্রামে যতক্ষণ পর্যন্ত টেস্ট কন্ডিশন বা শর্তের মান সত্য থাকে, লুপ বডির আবর্তন ততক্ষণ পর্যন্ত চলতে থাকে। জাভা প্রোগ্রামে লুপ নির্বাহের জন্য ব্যবহৃত অন্যতম লুপ কন্ট্রোল স্টেটমেন্টসমূহ হলঃ

- for স্টেটমেন্ট (for statement)
- while স্টেটমেন্ট (while statement)
- do-while স্টেটমেন্ট (do-while statement)
- continue স্টেটমেন্ট (continue statement)

for স্টেটমেন্ট (for statement) ঃ জাভা প্রোগ্রামে কোন স্টেটমেন্ট দুই বা ততোধিক বার সম্পন্ন করার জন্য for স্টেটমেন্ট ব্যবহৃত হয়। সাধারণত কোন ভেরিয়েবল ব্যবহার করে for লুপের আবর্তন সংখ্যা গণনা করা হয়। এরূপ ভেরিয়েবলকে কাউন্টার ভেরিয়েবল বলা হয়।

সিনটেক্স (Syntax) ঃ for স্টেটমেন্টের ফরম্যাট দেয়া হল ঃ

```
CounterDecleration;
for ( CounterInitialization; Condition; CounterDecrement / Increment)
{
//Statement(s);
}
```

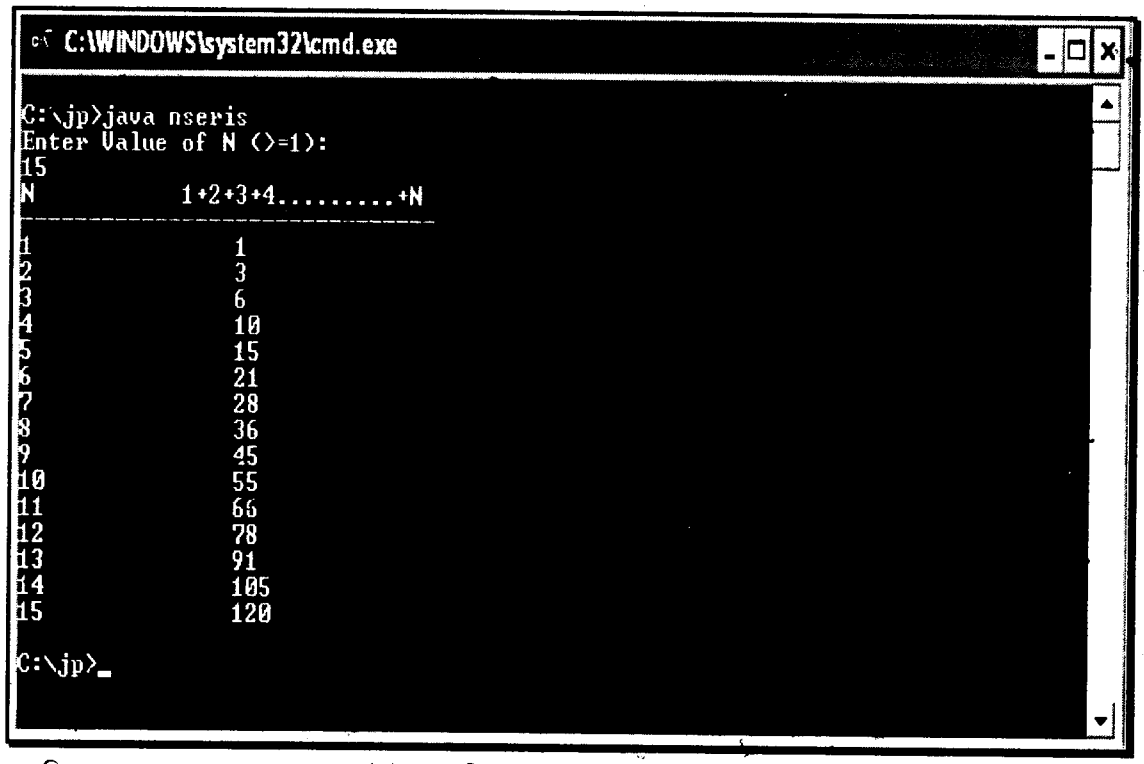
CounterDecleration অংশে উপযুক্ত ডাটা টাইপসহ কাউন্টার ভেরিয়েবল ঘোষণা করা হয়, CounterInitialization অংশে কাউন্টার ভেরিয়েবলের প্রারম্ভিক মান দেয়া হয়, Condition অংশে কাউন্টার ভেরিয়েবলের চূড়ান্ত মান কিংবা চূড়ান্ত মান নির্ধারণের শর্ত দেয়া হয় এবং CounterDecrement / Increment অংশে প্রতিবার আবর্তনে কাউন্টার ভেরিয়েবলের হ্রাস/বৃদ্ধির মান নির্ধারণ করা হয়। কাউন্টার ভেরিয়েবল চূড়ান্ত মানে না পৌঁছা পর্যন্ত কিংবা শর্ত সত্য থাকা পর্যন্ত for লুপের সাথে সংশ্লিষ্ট স্টেটমেন্ট সম্পাদিত হতে থাকে।

প্রোগ্রাম ঃ নিম্নে প্রথম N পর্যন্ত ধারাবাহিক সংখ্যার যোগফল প্রদর্শনের জন্য for স্টেটমেন্ট ব্যবহার করে একটি প্রোগ্রাম দেয়া হল।

```
import java.io.*;
public class nseris
{
public static void main(String args[])
{
int i,n,sum=0;
String s;
try
{
BufferedReader BR=new BufferedReader(new InputStreamReader(System.in));
```

```
System.out.println("Enter Value of N (>=1): ");
s=BR.readLine();
n=Integer.parseInt(s);
System.out.println("N   1+2+3+4.....+N");
System.out.println("-----------------------------");
for(i=1;i<=n;i++)
{
sum=sum+i;
System.out.println("+i+"\t\t"+sum);
}
}
catch(Exception E){}
}
}
```

আউটপুট ঃ



সি++ এর মত জাভা প্রোগ্রামে কাউন্টার ভেরিয়েবলকে for লুপের প্রথম বন্ধনীর মধ্যেও ঘোষণা করা যায়। আবার Condition এবং CounterDecrement / Increment অংশ for স্টেটমেন্টের প্রথম বন্ধনীর মধ্যে ঘোষণা না করে Statement(s) অংশেও ঘোষণা করা যায়।

প্রোগ্রাম ঃ নিম্নে প্রথম ১০টি সংখ্যার Fibonacci সিরিজ প্রদর্শনের জন্য for লুপ ব্যবহার করে আরও একটি প্রোগ্রাম দেয়া হল।

```
public class fibo
{
public static void main(String args[])
{
int f0=0,f1=1,f;
System.out.println("First 10 Fibonacci Nubers are:\n");
for(int i=1;i<=10;i++)
{
f=f0+f1;
System.out.print(" "+f+" ");
f1=f0;
f0=f;
}
}
}
```

প্রয়োজনে একটি for লুপের মধ্যে অপর এক বা একাধিক for লুপ ব্যবহার করা যায়। এরূপ for লুপকে নেস্টেড for লুপ এবং মধ্যবর্তী for লুপকে ইনার (Inner) for লুপ, বহিঃস্থ for লুপকে আউটার (Outer) for লুপ বলা হয়। আউটার for লুপের পূর্বে ইনার for লুপের কাজ শেষ হয়।

প্রোগ্রাম ঃ নেস্টেড for লুপ ব্যবহার করে ২৫ পর্যন্ত মৌলিক সংখ্যাগুলো বের করার একটি প্রোগ্রাম দেয়া হল।

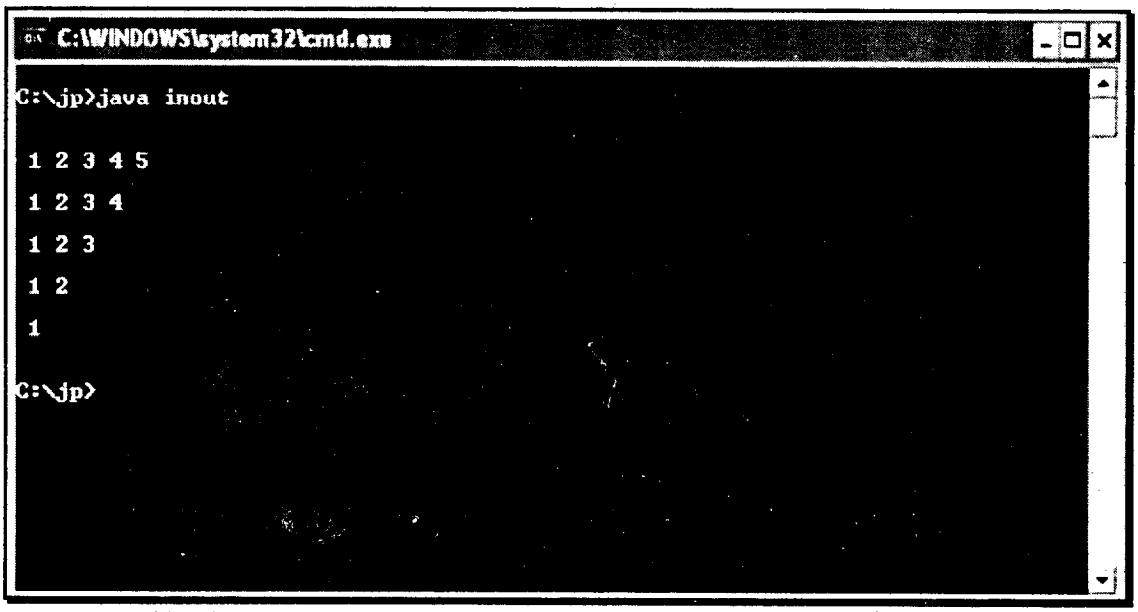
```
public class prime
{
public static void main(String args[])
{
int i,j;
System.out.println("Series of Prime number upto 25 is:\n");
for(i=2;i<=25;i++)
{
for (j=2;j<=i;j++)
{
if(i%j= =0)
break;
}
if(i= =j)
System.out.print(" "+i+" ");
}
}
}
```

প্রোগ্রাম ঃ নিম্নে ইনার এবং আউটার for লুপ ব্যবহার করে আরও একটি প্রোগ্রাম দেয়া হল।

```
public class inout
{
public static void main(String args[])
{
int i,j;
System.out.println("\n");
for(i=5;i>=1;i--)
{
for (j=1;j<=i;j++)
{
System.out.print(" "+j);
}

System.out.println("\n ");
}
}
}
```

আউটপুট ঃ



while স্টেটমেন্ট (While statement) ঃ জাভা প্রোগ্রামে শর্ত সাপেক্ষে দুই বা ততোধিক বার কোন স্টেটমেন্ট সম্পাদন করার জন্য while স্টেটমেন্ট ব্যবহৃত হয়। এটি অনেকটা for স্টেটমেন্টের বিকল্প হিসেবে ব্যবহৃত হয়। for স্টেটমেন্টের মত পূর্বে ঘোষিত কোন কাউন্টার ভেরিয়েবল ব্যবহার করে while স্টেটমেন্টের আবর্তন গণনা করা হয়।

সিনটেক্স (Syntax) ঃ নিম্নে while স্টেটমেন্টের ফরম্যাট দেয়া হল।

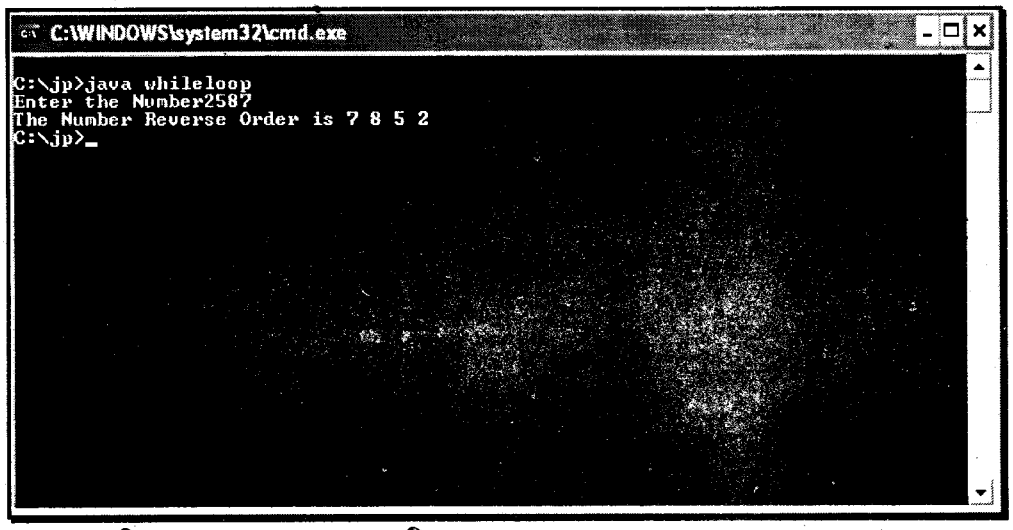
```
CounterDecleration;
CounterInitialization;
while(Condition)
{
Statement(s);
CounterDecrement/Increment;
}
```

CounterDecleration অংশে উপযুক্ত ডাটা টাইপসহ কাউন্টার ভেরিয়েবল ঘোষণা করা হয় এবং CounterInitialization অংশে তার প্রারম্ভিক মান নির্ধারণ করা হয়। Condition অংশে ইনডেক্স ভেরিয়েবলের চূড়ান্ত মান নির্ধারণের জন্য শর্ত দেয়া থাকে এবং CounterDecrement/Increment অংশে প্রতি আবর্তন কালে কাউন্টার ভেরিয়েবলের হ্রাস/বৃদ্ধির মান নির্ধারণ করা হয়। জাভা প্রোগ্রামে while লুপের বিকল্প হিসেবে do লুপ ব্যবহৃত হয়। তবে while স্টেটমেন্ট এবং do স্টেটমেন্টের পার্থক্য হল ঃ while স্টেটমেন্টে প্রথমে সংশ্লিষ্ট শর্ত পরীক্ষা করা হয়। শর্ত সত্য থাকা সাপেক্ষে while স্টেটমেন্টের সাথে সংশ্লিষ্ট স্টেটমেন্ট সম্পাদিত হয় , শর্ত সত্য না হলে while স্টেটমেন্টের সাথে সংশ্লিষ্ট স্টেটমেন্ট সম্পাদিত হয় না। কিন্তু do স্টেটমেন্টের ক্ষেত্রে প্রথমে do লুপের সাথে সংশ্লিষ্ট স্টেটমেন্ট একবার সম্পাদিত হয়। অতঃপর while লুপের সাথে সংশ্লিষ্ট শর্ত পরীক্ষা করা হয়। এই শর্ত সত্য থাকা সাপেক্ষে do লুপ স্টেটমেন্টের সাথে সংশ্লিষ্ট স্টেটমেন্ট আবর্তিত হয়। অর্থাৎ শর্ত সত্য না হলেও do লুপের সাথে সংশ্লিষ্ট স্টেটমেন্ট ন্যূনতম একবার সম্পাদিত হয়। এজন্য যখন ন্যূনতম একবার কোন কাজ করা আবশ্যক, সেক্ষেত্রে do লুপ ব্যবহার করা আবশ্যক।

প্রোগ্রাম ঃ নিম্নে do লুপ স্টেটমেন্ট ব্যবহার করে কোন সংখ্যার অংকগুলো বিপরীতক্রমে প্রদর্শনের একটি প্রোগ্রাম দেয়া হল।

```
import java.io.*;
public class whileloop
{
public static void main(String args[])throws IOException
{
int n,r;
String s;
BufferedReader BR=new BufferedReader(new InputStreamReader(System.in));
System.out.print("Enter the Number");
s=BR.readLine();
n=Integer.parseInt(s);
System.out.print("The Number Reverse Order is");
do
{
r=n%10;
System.out.print(" "+r);
n=n/10;
}while(n!=0);
}
}
```

আউটপুট ঃ



প্রয়োজনে একটি do লুপের মধ্যে অপর একটি while লুপ, do লুপ বা for লুপ ব্যবহার করা যেতে পারে। এরূপ মধ্যবর্তী লুপকে নেস্টেড লুপ বলা হয়। আবার Condition অংশ দুই বা ততোধিক শর্তবিশিষ্ট কোন রিলেশনাল এক্সপ্রেশন হতে পারে।

প্রোগ্রাম ঃ নিম্নে A থেকে z পর্যন্ত বড় হাতের অক্ষরগুলো প্রদর্শনের জন্য একটি প্রোগ্রাম দেয়া হল।

```
public class dis
{
public static void main(String args[])
{
char ch='A';
do
{
System.out.print(" "+ch+" ");
ch++;
}while((ch>='B') && (ch<='Z'));
}
}
```

continue স্টেটমেন্ট (continue statement) ঃ লুপের মধ্যে শর্তযুক্ত অথবা শর্তবিহীনভাবে কোন স্টেটমেন্ট বা লুপের পুনরাবৃত্তি করার জন্য continue স্টেটমেন্ট ব্যবহার হয়।

সিনটেক্স (Syntax) ঃ continue স্টেটমেন্টের সিনটেক্স হল

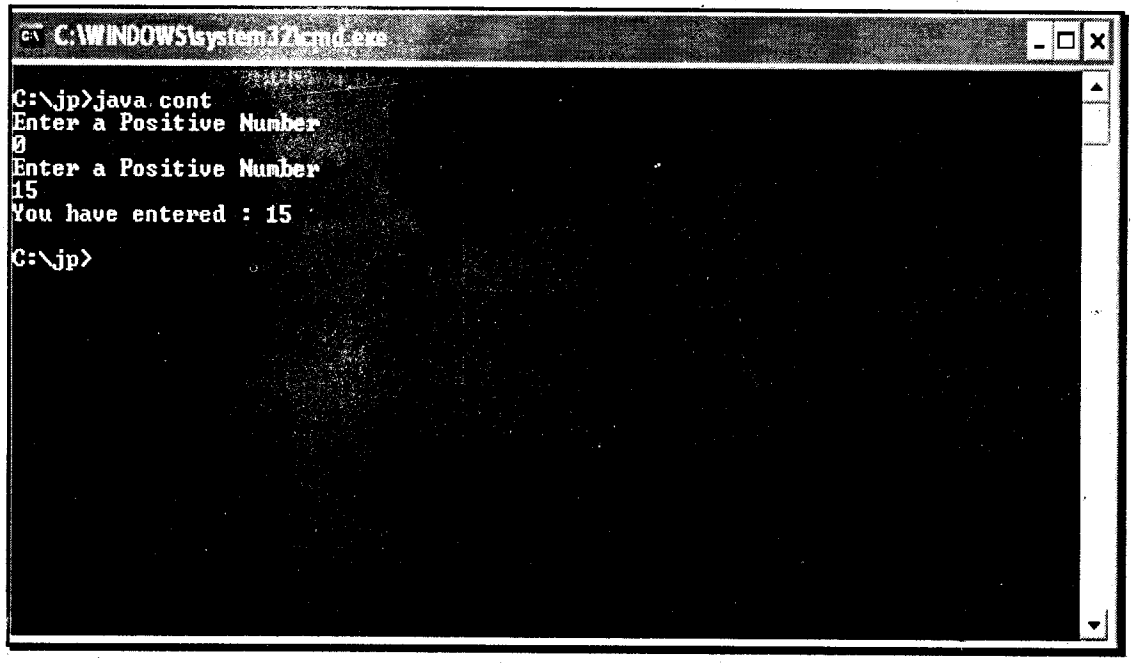
```
continue;
```

continue স্টেটমেন্ট প্রোগ্রাম পয়েন্টারকে পূর্ববর্তী স্টেটমেন্ট বা লুপের প্রারম্ভে স্থানান্তর করে। continue স্টেটমেন্ট if,else if,for, while ইত্যাদি ছাড়া সরাসরি কাজ করতে পারে না। তবে শর্তবিহীন continue স্টেটমেন্ট অসীম লুপের সৃষ্টি করে। এজন্য সাধারণত if, else....if স্টেটমেন্টের সাথে সম্পর্কিত শর্ত সাপেক্ষে কোন লুপের পুনরাবৃত্তি করার জন্য continue স্টেটমেন্ট ব্যবহৃত হয়। সে ক্ষেত্রে শর্তের মান সত্য হলে continue স্টেটমেন্ট কার্যকরী হয়, অন্যথায় কম্পাইলার continue স্টেটমেন্ট উপেক্ষা করে পরবর্তী স্টেটমেন্ট নির্বাহ করে।

প্রোগ্রাম ঃ নিম্নে continue স্টেটমেন্ট ব্যবহার করে একটি প্রোগ্রাম দেয়া হল।

```
import java.io.*;
public class cont
{
public static void main(String args[]) throws IOException
{
int n=0;
String s;
for(;;)
{
BufferedReader BR =new BufferedReader (new InputStreamReader (System.in));
System.out.println("Enter a Positive Number");
s=BR.readLine();
n=Integer.parseInt(s);
if(n<=0)
continue;
else
break;
}
System.out.println("You have entered : " + n);
}
}
```

আউটপুট ঃ

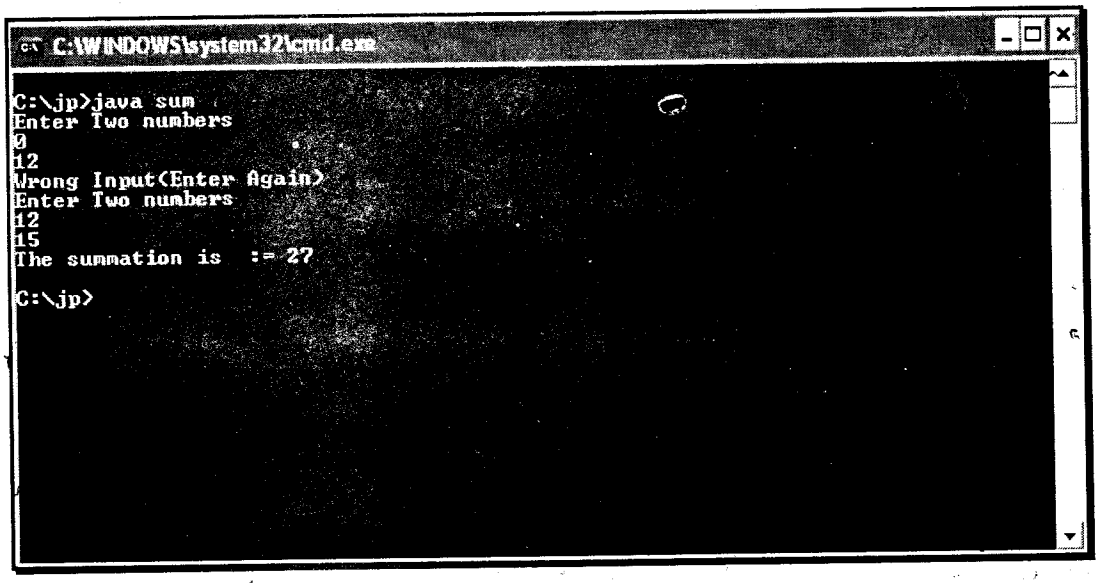


প্রোগ্রাম ঃ

নিম্নে continue এবং break স্টেটমেন্ট ব্যবহার করে দুটি সংখ্যার যোগফল প্রদর্শনের একটি প্রোগ্রাম দেয়া হলঃ

```
import java.io.*;
public class sum
{
public static void main(String args[]) throws IOException
{
int x,y,z;
String s1,s2;
loop1:
for(;;)
{
BufferedReader BR=new BufferedReader(new InputStreamReader(System.in));
System.out.println("Enter Two numbers");
s1=BR.readLine();
s2=BR.readLine();
x=Integer.parseInt(s1);
y=Integer.parseInt(s2);
if(x<=0 || y<=0)
{
System.out.print("Wrong Input(Enter Again)");
continue loop1;
}
else
{
z=x+y;
System.out.println("The summation is := " + z);
break loop1;
}
}
}
}
```

আউটপুট ঃ



৬.৬ জাভা ভাষায় অ্যারে (Array in Java) ঃ

একটি সাধারণ ভেরিয়েবলের নামের আওতায় মেমোরিতে পরপর সংরক্ষিত একই টাইপের কতগুলো ডাটার সমষ্টিকে অ্যারে (Array) বলা হয়। অ্যারে হল একই টাইপের কতগুলো ভেরিয়েবলের সেট। অ্যারে ভেরিয়েবল একই নামে, একই টাইপের একাধিক ডাটা সংরক্ষণে সক্ষম। অ্যারে একটি ডিরাইভড ডাটা টাইপ। সাধারণত ভেরিয়েবল ঘোষণার মত ব্যবহারের পূর্বে ডাটা টাইপসহ অ্যারে ভেরিয়েবল ঘোষণার প্রয়োজন হয়।

সিনটেক্স (Syntax) ঃ জাভা ভাষায় অ্যারে ঘোষণার ফরম্যাট হল ঃ

```
DataType ArrayName[ ];
ArrayName =  new DataType [ ArraySize ];
```

আবার উপরোক্ত দুই স্টেটমেন্টকে একত্রে এভাবেও লেখা যায় ঃ

```
DataType ArrayName[ ]=new DataType [ ArraySize ];
```

এখানে, DataType যে কোন বিল্ট-ইন বা কাস্টম ডাটা টাইপ, ArrayName প্রোগ্রামার কর্তৃক দেয় অ্যারে ভেরিয়েবলের যে কোন বৈধ নাম এবং ArraySize পূর্ণ সংখ্যায় প্রকাশিত কোন মান। অ্যারে ভেরিয়েবলের নাম একটি আইডেন্টিফায়ার। সুতরাং ভেরিয়েবলের যে কোন বৈধ নাম অ্যারের নাম হিসেবে দেয়া যাবে। অ্যারের নাম সংলগ্ন তৃতীয় বন্ধনীর '[ ]' মধ্যে অ্যারে সাইজ নির্ধারণ করা হয় যা অ্যারে ভেরিয়েবলে সর্বোচ্চ ডাটার সংখ্যা নির্দেশ করে; এই সংখ্যাকে অ্যারে ইনডেক্স (index) বলা হয় এবং অ্যারের প্রতিটি স্বতন্ত্র ভেরিয়েবলকে আলাদাভাবে অ্যারে উপাদান (Array element) বলা হয়।

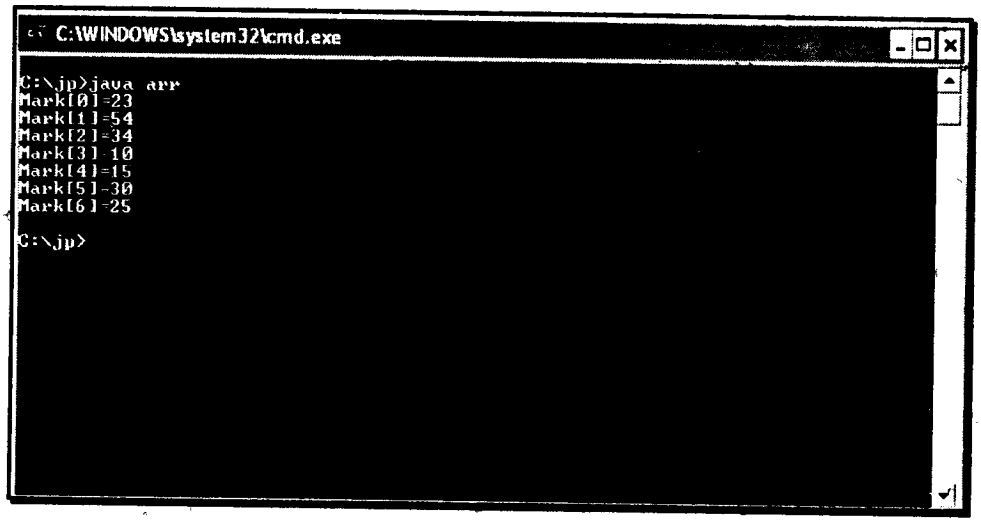
উদাহরণঃ

```
char name[ ]=new char[20];
int roll [ ]=new int[10];
float mark[ ]=new float[5];
```

নিম্নে অ্যারে ব্যবহার করে একটি প্রোগ্রাম দেয়া হল ঃ

```
public class array
{
public static void main(String args[ ])
{
int mark[ ]={23,54,34,10,15,30,25};
for(int i=0;i<=6;i++)
{
System.out.println("Mark[" +i+"]="+mark[i]);
}
}
}
```

আউটপুট ঃ



অ্যারে ব্যবহারের সুবিধা বা কারণ ঃ

- একাধিক ডাটার (একই ধরনের) জন্য একটি নাম ব্যবহার করা হয় বলে প্রোগ্রামারের সময় কম অপচয় হয়।
- একটি ভেরিয়েবল নিয়ে প্রোগ্রামার দক্ষতার সাথে কাজ করতে পারে।

অ্যারের মাত্রার বর্ণনা (Dimension of array) ঃ মাত্রা বা dimension এর উপর ভিত্তি করে অ্যারেকে নিম্নলিখিত ভাগে ভাগ করা যায় ঃ

- একমাত্রিক (One dimensional array)
- দ্বিমাত্রিক (Two dimensional array)
- বহুমাত্রিক (Multi dimensional array)

একমাত্রিক অ্যারে (One dimensional array) ঃ একটি মাত্র subscript নিয়ে গঠিত অ্যারেকে এক মাত্রিক অ্যারে বলে। এর গঠন ঃ

```
DataType ArrayName [size];
```

উদাহরণ ঃ
```
int i[10];
float a[10];
char ch[20];
double b[5];
```

আমরা যদি পাঁচটি সংখ্যা (20, 45, 30, 10, 35) কে integer type এর একটি array তে দেখাই -

```
int number[5];
```

number(0)	20
number(1)	45
number(2)	30
number(3)	10
number(4)	35

দ্বিমাত্রিক অ্যারে ঃ দুটি subscript নিয়ে গঠিত অ্যারেকে দ্বিমাত্রিক অ্যারে বলে। একে অনেক সময় matrixও বলা হয়। এর গঠন ঃ

```
DataType ArrayName [row_size][col_size];
```

যেমন– int student[3][4];

60 [0][0]	80 [0][1]	75 [0][2]	62 [0][3]
40 [1][0]	90 [1][1]	82 [1][2]	34 [1][3]
25 [2][0]	10 [2][1]	35 [2][2]	42 [2][3]

বহুমাত্রিক অ্যারে ঃ তিন বা ততোধিক subscript নিয়ে গঠিত অ্যারেকে বহুমাত্রিক অ্যারে বলে।

এর গঠন ঃ

```
DataType ArrayName [s1][s2] .. ... [sn];
```

উদাহরণ ঃ
```
int s[2][3][4];
float table[5][6][4][5];
```

অ্যারে ইনিশিয়ালাইজেশন (Array initialization) ঃ

একমাত্রিক অ্যারে (One dimensional array) ঃ

সাধারণ গঠন static DataType ArrayName[size]={list of values }; valueগুলোকে পরস্পর কমা দ্বারা পৃথক করা হয়।

- সাধারণত : static int number[3]={5,0,7};
- Array এর সাইজ উল্লেখ না করলেও চলে।

```
static int counter[]={1,3,2,5,7,8,12,54};
static char name[]={'D', 'H', 'A', 'K', 'A');
```

দ্বিমাত্রিক অ্যারে (Two dimensional array) ঃ

- একমাত্রিক অ্যারের মত : static int table[2][3]={1,2,3,4,5,6};
- Row হিসেবে আলাদা করে : static int table[2][3]={{1,2,3},{4,5,6}};
- Matrix এর মত করে : static int table[2][3]={{1,2,3},{4,5,6}};
- সব element শূন্য দ্বারা initialize করতে , শুধুমাত্র row এর প্রথাটি initialize করলে হয় :

প্রোগ্রাম ঃ static int table[2][3]={{0},{0}};

```
import java.io.*;
public class input
{
public static void main(String args[] ) throws IOException
{
int roll[]=new int [5];
String s[]=new String[5];
for(int i=0; i<5;i++)
{
System.out.print("\n Enter Roll["+i+"]");
BufferedReader BR=new BufferedReader (new InputStreamReader(System.in));
s[i]=BR.readLine();
roll[i]=Integer.parseInt(s[i]);
}
for(int i=0;i<5;i++)
{
System.out.print("\n Roll["+i+"]="+roll[i]);
}
}
}
```

৬.৭ অপারেটর, কন্ট্রোল স্টেটমেন্ট ও অ্যারে ব্যবহার করে প্রোগ্রাম (Programs using Operators, Control Statements and Array) ঃ

উক্ত অধ্যায়ের পূর্বের টপিক্সগুলোতে এবং বইয়ের ব্যবহারিক অংশে প্রোগ্রাম দেয়া আছে।

অনুশীলনী-৬

অতি সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। ডাটা টাইপ বলতে কী বুঝায়?

উত্তরঃ ডাটার মান, ধরন এবং মেমোরি স্পেস সংরক্ষণের ভিত্তিতে প্রোগ্রামে ব্যবহৃত ডাটাসমূহকে স্পেসিফাই করাকে ডাটা টাইপ বলে। যেমন- int, char, float, double ইত্যাদি।

২। প্রিমিটিভ ও নন-প্রিমিটিভ ডাটা টাইপ কাকে বলে?

উত্তরঃ **প্রিমিটিভ ডাটা টাইপ ঃ** জাভায় যে সকল বিল্ট-ইন ডাটা টাইপ ব্যবহৃত হয় তাদেরকে প্রিমিটিভ ডাটা টাইপ বলে। যেমন- int, char, float, double, boolean, byte, short, long ইত্যাদি।

নন-প্রিমিটিভ ডাটা টাইপ ঃ ইউজার কর্তৃক ডিফাইনকৃত অথবা এক বা একাধিক প্রিমিটিভ ডাটা টাইপের সমন্বয়ে গঠিত ডাটা টাইপকে নন-প্রিমিটিভ ডাটা টাইপ বলে। যেমন- অ্যারে, স্ট্রিং, ক্লাস, ইন্টারফেস ইত্যাদি।

৩। জাভায় কয় ধরনের ও কী কী বিল্ট ইন ডাটা টাইপ ব্যবহৃত হয়?

উত্তরঃ জাভায় মোট আট ধরনের বিল্ট ইন ডাটা টাইপ ব্যবহৃত হয়। যথা ঃ int, char, float, double, boolean, byte, short, long ইত্যাদি।

৪। ভেরিয়েবলের ডাইনামিক ইনিশিয়ালাইজেশন কী?

উত্তরঃ ভেরিয়েবলকে ডিক্লেয়ার করার সময় একই সাথে এর মান নির্ধারণ করে দেয়াকে ভেরিয়েবলের ডাইনামিক ইনিশিয়ালাইজেশন বলে। যেমন- int length = 10; float PI = 3.1416; ইত্যাদি।

৫। স্কোপ ও লাইফ টাইম বলতে কী বুঝায়?

উত্তরঃ কোন ভেরিয়েবলকে প্রোগ্রামের কোথায় ব্যবহার করা যাবে কিংবা কোথায় ব্যবহার করা যাবে না তা নির্ধারণ করে দেয়ার প্রক্রিয়াই হচ্ছে ঐ ভেরিয়েবলের স্কোপ (scope)

কোন ভেরিয়েবল তার মধ্যে রক্ষিত ভ্যালু (Value) কে যত সময় সংরক্ষণ করতে পারে তাই হল ঐ ভেরিয়েবলের লাইফটাইম (life time)।

৬। জাভা ভাষায় ভেরিয়েবল স্কোপ নির্ধারণের জন্য কয় ধরনের ও কী কী টাইপ মডিফায়ার ব্যবহৃত হয়?

উত্তরঃ জাভায় ভেরিয়েবলের স্কোপ নির্ধারণের জন্য ৭ ধরনের টাইপ মডিফায়ার ব্যবহৃত হয়। যথা ঃ abstract, final, native, static, synchronized, transient, volatile ইত্যাদি।

৭। ইউনারি ও বাইনারি অপারেটর বলতে কী বুঝায়?

উত্তরঃ **ইউনারি অপারেটর (Unary Operator) ঃ** যে সকল অপারেটর একটি মাত্র অপারেন্ড নিয়ে কাজ করে তাদেরকে ইউনারি অপারেটর বলে। যেমন ঃ +, − , + +, − − ইত্যাদি।

উদাহরণ ঃ Value z = + value 1, counter − − ইত্যাদি।

বাইনারি অপারেটর (Binary operator) ঃ যে সকল অপারেটর একই সাথে দুইটি অপারেন্ড নিয়ে কাজ করে তাদেরকে বাইনারি অপারেটর বলে। যেমন ঃ +, − , %, <, >, = =, & &, ?: ইত্যাদি।

উদাহরণ ঃ OP3 = OP1 + OP2, OP1< 10, ! OP1 ইত্যাদি।

৮। এক্সপ্রেশন কী?

উত্তরঃ কতগুলো Operand, operator এবং Constant এর সামঞ্জস্যপূর্ণ উপস্থাপনকে এক্সপ্রেশন বলে। যেমন, C = a + b একটি এক্সপ্রেশন।

৯। পাঙ্কচুয়েটর (Punctuator) কী?

উত্তরঃ জাভা প্রোগ্রাম লেখার সময় বিভিন্ন টোকেনের মধ্যে পার্থক্য নির্দেশ বা সমন্বয় সাধন করতে কিংবা প্রোগ্রামের সৌন্দর্য বাড়াতে কতকগুলো ক্যারেক্টার বা বিরাম চিহ্ন ব্যবহৃত হয়। এগুলোকে পাঙ্কচুয়েটর বলা হয়। যেমন– স্পেস, কমা, কোটেশন, সেমিকোলন ইত্যাদি।

১০। স্টেটমেন্ট (Statement) কী?

উত্তরঃ প্রতিটি জাভা প্রোগ্রাম কতকগুলো এক্সপ্রেশনের সমন্বয়ে গঠিত। এ এক্সপ্রেশনসমূহ অর্থপূর্ণ অথবা অর্থহীনও হতে পারে। প্রোগ্রামে ব্যবহৃত অর্থপূর্ণ এক্সপ্রেশনকে স্টেটমেন্ট বলে।

জাভা প্রোগ্রামে স্টেটমেন্টসমূহকে দুই ভাগে ভাগ করা হয়–

(i) সিম্পল স্টেটমেন্ট (Simple Statement)

(ii) কম্পাউন্ড স্টেটমেন্ট (Compound Statement)

১১। জাভায় ব্যবহৃত স্পেশাল অপারেটরসমূহের নাম লেখ।

উত্তরঃ জাভায় ব্যবহৃত স্পেশাল অপারেটরগুলো হল ঃ

- কমা অপারেটর (Comma operator)
- ইন্সট্যান্স অফ অপারেটর (instance of operator)
- ডট অপারেটর (Dot (.) operator)

১২। ইন্সট্যান্স অপারেটরের ব্যবহার উল্লেখ কর।

উত্তরঃ কোন অবজেক্ট কোন ক্লাসের আওতাভুক্ত কিনা তা নির্ধারণের জন্য instance of অপারেটর ব্যবহৃত হয়। কোন অবজেক্ট কোন ক্লাসের আওতাভুক্ত (instance) হলে instance of অপারেটর 'True' আর না হলে 'False' মান রিটার্ণ করে।

১৩। ডট অপারেটরের ব্যবহার লেখ।

উত্তরঃ কোন ক্লাসের ইন্সট্যান্স ভেরিয়েবল এবং মেথড এক্সেস (Access) করার জন্য (.) ডট অপারেটর ব্যবহৃত হয়।

১৪। Continue স্টেটমেন্ট ব্যবহার করা হয় কেন? এর ফরম্যাট লেখ।

উত্তরঃ জাভা প্রোগ্রামে শর্তযুক্ত অথবা শর্তবিহীনভাবে কোন স্টেটমেন্ট বা লুপের পুনরাবৃত্তি করার জন্য continue স্টেটমেন্ট ব্যবহৃত হয়।

continue স্টেটমেন্টের ফরম্যাট হল continue;

১৫। জাভায় অ্যারে ব্যবহারের সুবিধা বা কারণ কী?

উত্তরঃ অ্যারে ব্যবহারের সুবিধা বা কারণ হল ঃ

- একাধিক ডাটার (একই ধরনের) জন্য একটি নাম ব্যবহার করা হয় বলে প্রোগ্রামারের সময় কম অপচয় হয়।
- একটি ভেরিয়েবল নিয়ে প্রোগ্রামার দক্ষতার সাথে কাজ করতে পারে।

১৬। কম্পাউন্ড স্টেটমেন্ট কী?

উত্তরঃ দুই বা ততোধিক সিম্পল স্টেটমেন্টকে দ্বিতীয় বন্ধনীর মধ্যে লিখলে তাকে কম্পাউন্ড স্টেটমেন্ট বলে। যেমন–

```
{
int length = 10;
int width = 20;
int Area;
Area = legth + width;
}
```

১৭। শর্টহ্যান্ড অ্যাসাইনমেন্ট অপারেটর কাকে বলে?

উত্তরঃ জাভা প্রোগ্রামে অ্যাসাইনমেন্ট অপারেটরের সাথে গাণিতিক অপারেটর ব্যবহার করার রীতিকে শর্টহ্যান্ড অ্যাসাইনমেন্ট অপারেটর বলা হয়। যেমন ঃ x = x + 1 কে শর্টহ্যান্ড অপারেটরের সাহায্যে x + = 1 আকারে লেখা হয়।

সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। **জাভা ডাটা টাইপ হায়ারার্কী দেখাও।**

উত্তর সংকেত ঃ ৬.১ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **জাভায় ব্যবহৃত ডাটা টাইপের সাইজ ও রেঞ্জ উল্লেখ কর।**

উত্তর সংকেত ঃ ৬.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **জাভা ভেরিয়েবল নামকরণের নিয়মাবলি লেখ।**

উত্তর সংকেত ঃ ৬.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। **ক্লাস ভেরিয়েবল ও অটোমেটিক ভেরিয়েবলের সংজ্ঞা লেখ।**

উত্তর সংকেত ঃ ৬.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। **ইনক্রিমেন্ট ও ডিক্রিমেন্ট অপারেটরের পার্থক্য লেখ।**

উত্তর সংকেত ঃ ৬.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। **কন্ডিশনাল অপারেটরের ব্যাখ্যা দাও।**

উত্তর সংকেত ঃ ৬.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৭। **if-else স্টেটমেন্টের ফরম্যাট লেখ।**

উত্তর সংকেত ঃ ৬.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৮। **else-if স্টেটমেন্টের কারণ ব্যাখ্যা কর।**

উত্তর সংকেত ঃ ৬.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৯। **switch স্টেটমেন্টের Syntax লেখ।**

উত্তর সংকেত ঃ ৬.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১০। **while ও do-while লুপ স্টেটমেন্টের মধ্যে পার্থক্য লেখ।**

উত্তর সংকেত ঃ ৬.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১১। **জাভায় অ্যারে ব্যবহার করে একটি প্রোগ্রাম লেখ।**

উত্তর সংকেত ঃ ৬.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

➥ *রচনামূলক প্রশ্নোত্তর ঃ*

১। **জাভায় ব্যবহৃত অপারেটরসমূহ বর্ণনা কর।**

উত্তর সংকেত ঃ ৬.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **জাভা ভাষার কন্ট্রোল স্টেটমেন্টসমূহ ব্যাখ্যা কর।**

উত্তর সংকেত ঃ ৬.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **1 + 2 + 3 + ... + N = ? সিজিরে যোগফল নির্ণয়ের জাভা প্রোগ্রাম লেখ।**

উত্তর সংকেত ঃ ৬.৫ নং অনুচ্ছেদ অথবা ব্যবহারিক অংশ দ্রষ্টব্য।

৪। **প্রথম ১০টি সংখ্যার Fibonacci সিরিজ প্রদর্শনের জাভা প্রোগ্রাম লেখ।**

উত্তর সংকেত ঃ ৬.৫ নং অনুচ্ছেদ অথবা ব্যবহারিক অংশ দ্রষ্টব্য।

৫। **25 পর্যন্ত সংখ্যাগুলোর মধ্যকার মৌলিক সংখ্যাগুলো নির্ণয়ের জাভা প্রোগ্রাম লেখ।**

উত্তর সংকেত ঃ ৬.৫ নং অনুচ্ছেদ অথবা ব্যবহারিক অংশ দ্রষ্টব্য।

৬। **কোন একটি সংখ্যা ইনপুট দিয়ে তার ডিজিটগুলোকে রিভার্স অর্ডারে সাজানোর জাভা প্রোগ্রাম লেখ।**

উত্তর সংকেত ঃ ৬.৫ নং অনুচ্ছেদ অথবা ব্যবহারিক অংশ দ্রষ্টব্য।

৭। **কোন সংখ্যার ফ্যাক্টরিয়াল মান প্রদর্শনের জন্য জাভা প্রোগ্রাম লেখ।**

উত্তর সংকেত ঃ ৬.৫ নং অনুচ্ছেদ অথবা ব্যবহারিক অংশ দ্রষ্টব্য।

৮। **switch স্টেটমেন্ট ব্যবহার করে একটি প্রোগ্রাম লেখ।**

উত্তর সংকেত ঃ ৬.৫ নং অনুচ্ছেদ অথবা ব্যবহারিক অংশ দ্রষ্টব্য।

৯। **continue ও break স্টেটমেন্ট ব্যবহার করে দুটি সংখ্যার যোগফল নির্ণয়ের জাভা প্রোগ্রাম লেখ।**

উত্তর সংকেত ঃ ৬.৫ নং অনুচ্ছেদ অথবা ব্যবহারিক অংশ দ্রষ্টব্য।

ক্লাস, অবজেক্ট, মেথড ও কন্সটাক্টর
Class, Object, Method and Constructor in Java

অধ্যায় ০৭

৭.০ ভূমিকা (Introduction) ঃ

সাধারণত বাস্তব জগতের সমস্যা সমাধানের জন্য প্রোগ্রাম লেখা হয়। যেমন– একটি কোম্পানির কর্মচারীদের রেকর্ড সংরক্ষণ বা তদের বেতন, বোনাস ইত্যাদির হিসাব, কোন শিক্ষা প্রতিষ্ঠানের শিক্ষার্থীদের পরীক্ষার নম্বর সংরক্ষণ এবং ফলাফল প্রস্তুতকরণ ইত্যাদি। এরূপ সমস্যা বিল্ট-ইন ডাটা টাইপ ভেরিয়েবল ব্যবহার করে সমাধান সম্ভব। তবে সমস্যার ধরন অনুযায়ী নিজস্ব ডাটা টাইপ ঘোষণার পাশাপাশি ডাটার গোপনীয়তা, স্বাচ্ছন্দ্য ইত্যাদিও গুরুত্বপূর্ণ বিষয়। ক্লাস (Class) সি++ , জাভা তথা যে কোন অবজেক্ট অরিয়েন্টেড প্রোগ্রাম ভাষার একটি গুরুত্বপূর্ণ বৈশিষ্ট্য। ক্লাস একটি ইউজার-ডিফাইন্ড বা ব্যবহারকারী নির্ধারিত ডাটা টাইপ, যা প্রোগ্রামে বিল্ট-ইন ডাটা টাইপের মত কাজ করে। একটি ক্লাসে বিল্ট-ইন ডাটা টাইপের কয়েকটি ভেরিয়েবল এবং ঐ ভেরিয়েবলকে এক্সেস করার জন্য কতগুলো ফাংশন থাকে। ক্লাসের ভেরিয়েবলগুলোকে মেম্বার ভেরিয়েবল বা ফিল্ড, ফাংশনগুলোকে মেম্বার ফাংশন বা মেথড এবং ক্লাস টাইপ ভেরিয়েবলকে অবজেক্ট বলা হয়। প্রোগ্রামে প্রথমে কোন ক্লাস ঘোষণা করা হয়, অতঃপর ঐ ক্লাস টাইপের যে কোন সংখ্যক ভেরিয়েবল (অবজেক্ট) ঘোষণা করা যায়। প্রোগ্রাম নির্বাহের সময় কম্পাইলার অবজেক্টসমূহের জন্য স্বয়ংক্রিয়ভাবে প্রয়োজনীয় মেমোরি স্পেস বরাদ্দ করে এবং প্রোগ্রাম নির্বাহের শেষে বরাদ্দকৃত স্পেস খালি করে দেয়। মূলত কনস্ট্রাক্টর এবং ফাইনালাইজার মেথডের মাধ্যমে এই কাজটি হয়। ক্লাসের কোন ডাটা মেম্বারকে প্রাইভেট ঘোষণার মাধ্যমে বাইরের অন্য কোন ক্লাস বা মেথডের হস্তক্ষেপ থেকে রক্ষা করা যায়। ফলে সিস্টেমে ব্যবহৃত ডাটার গোপণীয়তা ও বিশ্বস্ততা বৃদ্ধি পায়।

৭.১ জাভা ভাষায় ক্লাস ও অবজেক্ট ডিক্লারেশন (Declaration (Syntax) of Class and Objects in Java) ঃ

ক্লাস (Class) ঃ ক্লাস একটি ইউজার-ডিফাইন্ড ডাটা টাইপ। একটি ক্লাসে কিছু ভেরিয়েবল এবং ঐ ভেরিয়েবলকে এক্সেস করার জন্য কিছু ফাংশন থাকে। ক্লাসের ভেরিয়েবলগুলোকে মেম্বার ভেরিয়েবল (member variable) বা ডাটা মেম্বার (data member) এবং মেম্বার ভেরিয়েবলগুলোর সাথে সম্পর্কিত ফংশনগুলোকে মেম্বার ফাংশন (member function) বা ফাংশন মেম্বার (function member) বা মেথড (method) বলা হয়। মেম্বার ভেরিয়েবল এবং মেম্বার ফাংশনসমূহকে একযোগে ক্লাস মেম্বার (class member) বলা হয়। একটি ক্লাসে মেম্বার হিসেবে বৈধ ডাটা টাইপের যেকোন সংখ্যক ভেরিয়েবল বা অন্য কোন ক্লাস ব্যবহৃত হতে পারে। ক্লাসের ঘোষণার মাধ্যমে ভিন্ন ভিন্ন ডাটা টাইপের (বিল্ট-ইন কিংবা কাস্টম) কতগুলো ভেরিয়েবল এবং ঐ ভেরিয়েবলগুলোর সাথে সম্পর্কিত ফাংশনসমূহকে একটি একক ইউনিটে পরিণত করা হয়। এতে কোন ডাটা এবং ফাংশনকে প্রয়োজনে বাহিরের অন্য কোন মেথডের সরাসরি ব্যবহার থেকে রক্ষা করা যায়। ক্লাসের মাধ্যমে এমন একটি ইউজার-ডিফাইন্ড ডাটা টাইপ তৈরি করা যায় যা পুরোপুরিভাবে অন্যান্য বিল্ট-ইন ডাটা টাইপের মত ব্যবহার করা যায়।

ক্লাস ডিক্লারেশন (Class declaration) ঃ জাভা ভাষায় ক্লাস ডিক্লারেশন বলতে ক্লাসের মেম্বার ভেরিয়েবল এবং মেম্বার ফাংশনগুলোর ডিক্লারেশন ও ডেফিনিশন বুঝায়। ক্লাস ডিক্লারেশনের জন্য class কীওয়ার্ডের পরে ঐ ক্লাসের নাম দিতে হয়, ক্লাস বডি দ্বিতীয় বন্ধনী '{ }' দ্বারা সীমাবদ্ধ থাকে। উল্লেখ্য, সি++ এর মত জাভা ক্লাস সেমিকোলন (;) দিয়ে শেষ হয় না। ক্লাস বডিতে ক্লাসের মেম্বার ভেরিয়েবল এবং মেম্বার ফাংশনগুলো তাদের এক্সেসিবিলিটি মোডসহ উল্লেখ করা হয়।

সিনটেক্স (Syntax) ঃ ক্লাস ডিক্লারেশনের সাধারণ ফরম্যাট হলঃ

```
class ClassName
{
Member Variable Decleration;
Member Function Definition;
}
```

একটি ক্লাস ডিক্লারেশনের প্রধানত দুটি অংশ থাকে। যথাঃ

- মেম্বার ভেরিয়েবল ডিক্লারেশন (Member Variable Decleration)
- মেম্বার ফাংশন বর্ণনা (Member Function Definitiion)

ভেরিয়েবল ডিক্লারেশন অংশে উপযুক্ত ডাটা টাইপসহ ক্লাসের মেম্বার ভেরিয়েবলগুলোর ডিক্লারেশন থাকে। আর মেম্বার ফাংশন ডেফিনিশন অংশে ক্লাসের মেম্বার ভেরিয়েবলগুলোর মান নির্ধারণ করা হয় এবং সেগুলোর ব্যবহারের বিস্তারিত ব্যাখ্যা থাকে। উল্লেখ্য জাভায় মেম্বার ফাংশন বা মেথডসমূহ ক্লাসের মধ্যে ডিফাইন করতে হয় যা সি++ এ ক্লাসের ভিতর ও বাহির উভয় জায়গায়ই করা যায়।

উদাহরণঃ

```
class Student      // Class Name
{
int Roll;          // Member Variable Decleration
string Name;       // Member Variable Decleration
float Mark;        // Member Variable Decleration
int GetData()      // Member function Definition
{
//Body of the Method
}
void Display()     // Member function Definition
{
//Body of the Method
}
}                  // End of the class
```

অবজেক্ট ডিক্লারেশন (Class declaration) ঃ ক্লাস ডাটা টাইপের ভেরিয়েবলকে অবজেক্ট (Object) বলা হয়। ক্লাসের কোন অবজেক্ট ঘোষণা করা মাত্র সেই অবজেক্টের জন্য মেমোরিতে প্রয়োজনীয় জায়গা বরাদ্দ হয়। প্রোগ্রামে প্রথমে ক্লাস ঘোষণা করা হয়, অতঃপর ঐ ক্লাস টাইপের যে কোন সংখ্যক অবজেক্ট ঘোষণা করা যায়। মূলত ক্লাসের মেম্বার ভেরিয়েবলগুলোর মান নির্ধারণের জন্য অবজেক্ট ঘোষণা করা হয়। এজন্য অবজেক্ট ঘোষণাকে অন্য অর্থে ইনস্ট্যানশিয়েটিং (instantiating) ও বলা হয়। জাভা ভাষায় অবজেক্ট ঘোষণার জন্য new অপারেটর ব্যবহৃত হয়। new অপারেটর নির্দিষ্ট ক্লাসের একটি অবজেক্ট তৈরি করে এবং ঐ অবজেক্টে একটি রেফারেন্স (মেমোরি অ্যাড্রেস) ফেরত পাঠায়।

সিনটেক্স (Syntax) ঃ ক্লাসে অবজেক্ট ডিক্লারেশনের ফরম্যাট নিম্নরূপ ঃ

```
ClassName ObjectName;
ObjectName = new ClassName ( );
```

উদাহরণ ঃ

```
student S;
S = new student ( );
```

এখানে 'student' হল ClassName এবং 'S' হল ObjectName.

৭.১.১ Class মেম্বার এক্সেসিং (Accessing Class Members) ঃ

ক্লাস মেম্বার এক্সেসিং বলতে কোন ক্লাসের প্রাইভেট, পাবলিক ও প্রটেকটেড অংশে ডিক্লেয়ারকৃত মেম্বার ভেরিয়েবল ও মেম্বার ফাংশনসমূহকে এক্সেস (Access) করা বুঝায়। সুতরাং বলা যায় ক্লাস মেম্বার এক্সেস করার জন্য তিন ধরনের এক্সেসিবিলিটি মোড বা ভিজিবিলিটি (Visibility) মোড রয়েছে। যথা ঃ

- এক্সেসিং পাবলিক মেম্বার (Accessing public member)
- এক্সেসিং প্রাইভেট মেম্বার (Accessing private members)
- এক্সেসিং প্রটেকটেড মেম্বার (Accessing protected member)

এক্সেসিং পাবলিক মেম্বার (Accessing Public Members) ঃ ক্লাসের পাবলিক মেম্বারসমূহ (ভেরিয়েবল বা মেথড) ঐ ক্লাসের জন্য গ্লোবাল মেম্বার হিসেবে বিবেচিত হয়। সুতরাং সেগুলোকে ঐ ক্লাসের নিজস্ব (প্রাইভেট, প্রটেকটেড কিংবা পাবলিক) মেথড কিংবা ঐ ক্লাসের কোন অবজেক্টের সাথে ডট অপারেটর ব্যবহার করে সরাসরি মেইন মেথড দ্বারা এক্সেস করা যায়।

প্রোগ্রাম ঃ

```
class Student
{
public int Roll;
public String Name;
public float Mark;
public void GetData()
{
Roll=96002;
Name="Tahsin";
Mark=85.25f;
}
void Display()
{
System.out.println("Roll is : "+Roll);
System.out.println("Name is : "+Name);
System.out.println("Mark is : "+Mark);
}
}
public class friend
{
public static void main(String args[])
{
Student s=new Student();
s.GetData();
s.Display();
}
}
```

এক্সেসিং প্রাইভেট মেম্বার (Accessing private members) ঃ ক্লাসের প্রাইভেট মেম্বারসমূহ (ভেরিয়েবল বা ফাংশন) ঐ ক্লাসের জন্য লোকাল মেম্বার হিসেবে বিবেচিত হয়। সুতরাং সেগুলোকে ঐ ক্লাসের নিজস্ব (প্রাইভেট, প্রটেকটেড কিংবা পাবলিক) মেম্বার ফাংশন ছাড়া সরাসরি অন্য কোনভাবে এক্সেস করা যায় না। এমন কি ঐ ক্লাসের কোন অবজেক্টের সাথে ডট অপারেটর ব্যবহার করে কিংবা মেইন ফাংশনের মধ্যে স্টেটমেন্ট লিখে ক্লাসের প্রাইভেট মেম্বার এক্সেস করা যায় না।

প্রোগ্রাম ঃ

```
class Student
{
private int Roll;
private String Name;
private float Mark;
private void GetData()
{
Roll=96002;
Name="Tahsin";
Mark=85.25f;
}
void Display()
{
GetData();
System.out.println("Roll is : "+Roll);
System.out.println("Name is : "+Name);
System.out.println("Mark is : "+Mark);
}
}
public class priv
{
public static void main(String args[])
{
Student s=new Student();
s.Display();
}
}
```

এক্সেসিং প্রটেকটেড মেম্বার (Accessing protected members) ঃ ক্লাসের প্রটেকটেড মেম্বারের (ভেরিয়েবল/ফাংশন) এক্সেসিবিলিটি অনেকটা পাবলিক এবং প্রাইভেট মেম্বারের মাঝামাঝি। ক্লাসের প্রটেকটেড মেম্বারগুলো সাধারনত তার নিজস্ব কোন মেম্বার পাবলিক কিংবা প্রটেকটেড মেম্বার ফাংশন সহযোগে এক্সেস করা যায়। পাবলিক মেম্বার ফাংশনের মত ক্লাসের প্রটেকটেড মেম্বারগুলো তার নিজস্ব কোন অবজেক্টের সাথে ডট অপারেটর সহযোগে এক্সেস করা যায়। নিম্নে ক্লাসের প্রটেকটেড মেম্বার এক্সেসিং এর জন্য একটি নমুনা প্রোগ্রাম দেয়া হল ঃ

প্রোগ্রাম ঃ

```
class Student
{
protected int Roll;
protected String Name;
protected float Mark;
protected void GetData()
{
Roll=96002;
Name="Tonny";
Mark=80.5f;
}
protected void Display()
{
System.out.println("Roll is : "+Roll);
System.out.println("Name is : "+Name);
System.out.println("Mark is : "+Mark);
}
}
public class pro
{
public static void main(String args[])
{
Student s=new Student();
s.GetData();
s.Display();
}
}
```

৭.২ অবজেক্ট রেফারেন্স ভেরিয়েবল এসাইন করার প্রক্রিয়া (Assigning an Object Reference Variable) ঃ

জাভা কিংবা সি++ এর মত অবজেক্ট অরিয়েন্টেড ল্যাংগুয়েজে অবজেক্ট একটি অতি গুরুত্বপূর্ণ বিষয়। কারণ কোন ক্লাসের প্রাইভেট, পাবলিক ও প্রটেকটেড এই তিন অংশে যেসব মেম্বার ভেরিয়েবল কিংবা মেম্বার ফাংশন ডিক্লেয়ার ও ডিফাইন করা হয় সেসব মেম্বার ভেরিয়েবল ও মেম্বার ফাংশন (মেথড)সমূহকে অবজেক্ট রেফারেন্স ভেরিয়েবল ছাড়া এক্সেস করা যায় না। তাই প্রোগ্রামের প্রতিটি ক্লাসের জন্য আলাদা আলাদা অবজেক্ট রেফারেন্স ভেরিয়েবল প্রয়োজন হয়। অবজেক্ট রেফারেন্স ভেরিয়েবল এসাইন করার প্রক্রিয়া নিম্নবর্ণিত ধারাবাহিক কাজের সমষ্টি। যেমন-

- অবজেক্ট ডিক্লারেশন (Object declaration)
- অবজেক্ট সহযোগে মেম্বার ফাংশন কলিং (Member function calling with object).

অবজেক্ট ডিক্লারেশন (Object declaration) ঃ অবজেক্ট ডিক্লারেশনের পূর্বেই ক্লাস ডিক্লেয়ার ও মেম্বার ফাংশন (মেথড) বা ফাংশনসমূহ ডিফাইন করতে হবে। এভাবে main ( ) মেথডের মধ্যে সংশ্লিষ্ট ক্লাসের জন্য অবজেক্ট ডিক্লেয়ার করতে হবে। অবজেক্ট ডিক্লারেশন ফরম্যাট হল ঃ

```
ClassName ObjectName = new ClassName ( );
```

উদাহরণ ঃ

```
Student S = new Student ( );
```

এখানে ClassName হল- 'Student', আর ObjectName হল- 'S'।

অবজেক্ট সহযোগে মেম্বার ফাংশন কলিং (Member function calling with object) ঃ দ্বিতীয় পর্যায়ে, অবজেক্ট ডিক্লেয়ার করার পর পরই ক্লাসের মেম্বার ফাংশনসমূহ কল (Call) করতে হবে। মেম্বার ফাংশন কল করার জন্য ডট অপারেটর (.) ব্যবহার করা হয়। অবজেক্ট সহযোগে মেম্বার ফাংশন কল করার ফরম্যাট হল ঃ

```
ObjectName. FunctionName ( );
```

উদাহরণ ঃ

```
S. GetData ( );
```

এখানে 'S' হল ObjectName এবং 'GetData' হল সংশ্লিষ্ট ক্লাসের একটি FunctionName। ক্লাসে একাধিক ফাংশন (method) থাকলে প্রতিটি আলাদা আলাদাভাবে একই প্রক্রিয়ায় কল করতে হয়।

৭.৩ সিনটেক্সসহ মেথড (Method with Syntax) ঃ

মেথড (Method) ঃ সি বা সি++ ভাষায় যাকে ফাংশন বলে জাভা ভাষায় তাকে মেথড (method) বলে। কোন প্রোগ্রাম যখন অনেক বড় হয়, তখন সেই প্রোগ্রাম লেখা এবং ব্যবস্থাপনা (manage) করা অনেক কঠিন হয়ে পড়ে। আবার কখনো কখনো প্রোগ্রামে একই কাজ বার বার করতে হয় এবং এজন্য একই স্টেটমেন্ট বারবার লিখতে হয়। এসব সমস্যার সমাধানকল্পে প্রোগ্রামে মেথড ব্যবহার করা হয়।

সুতরাং মেথড (method) হচ্ছে একটি বড় প্রোগ্রামের একটি স্বাধীন ক্ষুদ্র অংশ (module) যার একটি নির্দিষ্ট নাম থাকে, যা এক বা একাধিক স্টেটমেন্টের সমন্বয়ে গঠিত এবং নির্দিষ্ট কোন সমস্যা সমাধানের জন্য ব্যবহৃত হয়। কোন ক্লাস মেথডের মাধ্যমেই অবজেক্টকে কোন কার্যসম্পাদনের ক্ষমতা প্রদান করে। অবজেক্ট তার ক্লাসের মেথডকে কল করে এবং ফলশ্রুতিতে ঐ মেথডটি execute হয়।

সিনটেক্স (Syntax) ঃ একটি সাধারণ মেথডের ফরম্যাট নিম্নরূপ ঃ

```
AccessModifer ReturnType MethodName (Argument list)
{
// Method body
}
```

উদাহরণ ঃ

```
public int add (int n, int m)
{int k = nt m;
return k;
}
```

অর্থাৎ একটি মেথডে ৫টি অংশ থাকে। যথা ঃ

- একসেস মডিফায়ার (Access modifier)
- রিটার্ন টাইপ (Returen type)
- মেথডের নাম (MethodName)
- আরগুমেন্ট লিস্ট (Argument list) ও
- মেথড বডি (Method body)।

একসেস মডিফায়ার ও আরগুমেন্ট লিস্ট অংশ দু'টি ঐচ্ছিক (Optional)। অর্থাৎ মেথড ডিক্লারেশনে একসেস মডিফায়ার ও আরগুমেন্ট লিস্ট অনেক সময় ব্যবহৃত হয় না।

একসেস মডিফায়ার (Access Modifier) ঃ

মডিফায়ার হল জাভা কীওয়ার্ড যা কম্পাইলারকে কোড, ডাটা অথবা ক্লাসের nature সম্পর্কে অবহিত করে। সাধারণত ক্লাস, ভেরিয়েবল এবং মেথডের নামের আগে মডিফায়ার ব্যবহৃত হয়। সবচেয়ে বেশি ব্যবহৃত এবং কমন (Common) মডিফায়ার হল একসেস মডিফায়ার (Access modifier)। মেথডে ব্যবহৃত একসেস মডিফায়ারগুলো হল ঃ

- প্রাইভেট (private)
- পাবলিক (public) ও
- প্রটেকটেড (protected) ইত্যাদি।

রিটার্ন টাইপ (ReturnType) ঃ কোন মেথডের আউটপুটের প্রকৃতির উপর ভিত্তি করে এর রিটার্ন টাইপ নির্ধারিত হয়। রিটার্ন টাইপ হিসাবে যে কোন ধরনের ডাটা টাইপ (int, char, float, double, byte) ইত্যাদি হতে পারে। যদি কোন মেথডের রিটার্ন টাইপ 'int' হয় তাহলে বুঝতে হবে এটি ইন্টিজার টাইপের আউটপুট প্রদান করবে। আর যদি রিটার্ন টাইপ 'void' হয় তবে বুঝতে হবে মেথডটি কোন আউটপুট প্রদান করবে না।

মেথডের নাম (MethodName) ঃ প্রতিটি মেথডের অবশ্যই একটি নাম দিতে হবে। এই নামটি দেয়ার ক্ষেত্রে জাভার আইডেন্টিফায়ার নামকরণের নিয়মানুযায়ী মেথডের যেকোন বৈধ নাম ব্যবহার করা যাবে। মেথডের নাম যে কোন দৈর্ঘ্যের হতে পারে।

আরগুমেন্ট লিস্ট (Argumentlist) ঃ 'Argument list' হল মেথডে ব্যবহৃত আরগুমেন্টের তালিকা। মেথডের প্রথম বন্ধনী (Parentheses '( )') এর মধ্যে কোন ভেরিয়েবল ডিক্লেয়ার করা হলে তাকে আরগুমেন্ট ভেরিয়েবল বলে। মেথডে কোন আরগুমেন্ট ভেরিয়েবল না লাগলে তবে প্রথম বন্ধনীর মধ্যে কিছু লেখা হয় না।

মেথড বডি (Method body) ঃ মেথড বডি হচ্ছে মেথডের সবচেয়ে গুরুত্বপূর্ণ অংশ। কারণ এতে মেথডের বিস্তারিত বর্ণনা থাকে। মেথড বডি কেমন হবে তা মেথডের ধরন বা সমস্যার উপর নির্ভর করে। সমস্যার উপর ভিত্তি করে এতে এক বা একাধিক স্টেটমেন্ট ব্যবহার করা হয়। এসব স্টেটমেন্টের মাধ্যমে সমস্যা সংক্রান্ত নানা Calculation, processing এবং manipulation করা হয়।

৭.৪ ক্লাসে মেথড সংযোজনের প্রক্রিয়া (Procedure of Adding Method to Class) ঃ

জাভা ভাষায় সবকিছুই ক্লাসের ভিতরে থাকবে। যদিও সি++ এ ক্লাসের বাইরে ভেরিয়েবল ডিক্লারেশনের সুযোগ আছে কিন্তু জাভায় ক্লাসের বাইরে কোন কিছুই (ভেরিয়েবল, মেথড ইত্যাদি) ডিক্লেয়ার বা ডিফাইন করার সুযোগ নেই। প্রতিটি ক্লাসে ডাটা বা ভেরিয়েবল থাকে এবং ভেরিয়েবলগুলোকে manipulate করার জন্য মেথড (method) ব্যবহার করা হয়। একটি ক্লাসে এক বা একাধিক মেথড ব্যবহার করা যেতে পারে।

ক্লাসে মেথড সংযোজন বা ব্যবহারের ক্ষেত্রে প্রধান ৪টি বিষয় বিবেচনা করতে হয়। এগুলোকে মেথডের উপাদান বলে। উপাদানগুলো হল ঃ

- মেথড ডেফিনিশন (Method definition)
- প্রটোটাইপ (Prototype)
- মেথড কলিং (Method calling)
- মেথডের রিটার্ন স্টেটমেন্ট (Methods return statement)

মেথড ডেফিনিশন (Method definition) ঃ মেথড ডেফিনিশন মেথডের প্রধান বিবেচ্য বিষয়। মেথডটি কি কাজ করবে এবং কিভাবে করবে তা এই অংশের মাধ্যমে কম্পাইলারকে জানিয়ে দেয়া হয়। মেথড ডেফিনিশনের ফরম্যাট হল ঃ

```
AccessModifier ReturnType MethodName (Argumentlist)
{
// method body
}
```

এখানে 'AccessModifier' হিসেবে private, public কিংবা protected কীওয়ার্ড ব্যবহৃত হয়। 'ReturnType'- যে কোন বৈধ ডাটা টাইপ (int, char, float, byte), 'MethodName- হিসেবে আইডেন্টিফায়ার নামকরণের নিয়মানুযায়ী ইউজার প্রদত্ত যে কোন নাম এবং 'Argument list' হিসেবে মেথডে ব্যবহৃত আরগুমেন্ট তালিকাকে বুঝায়। আর মেথড বডিতে (Method body) মেথডের বিস্তারিত বর্ণনা থাকে।

উদাহরণ ঃ

```
public int add (int n, int m)
{ int k = n + m;
return k;
}
```

মেথড ডেফিনিশন (Method definition) অংশটি অবশ্যই ক্লাসের ভিতরে থাকবে।

প্রটোটাইপ (Prototype) ঃ মেথডের প্রটোটাইপ মেথড বর্ণনার সাথে সম্পর্কিত। কোন মেথডের প্রটোটাইপ বলতে তার সঠিক নাম, রিটার্ন টাইপ, আরগুমেন্ট তালিকা, সুনির্দিষ্ট ফরম্যাটে তার ব্যবহারবিধি ইত্যাদি বুঝায়। বিল্ট-ইন মেথডসমূহের প্রটোটাইপ সংশ্লিষ্ট ক্লাসের সাথে সম্পর্কিত। এজন্য জাভা প্রোগ্রামে কোন বিল্ট-ইন মেথড ব্যবহার করলে প্রোগ্রামের শুরুতে অর্থাৎ ক্লাস বর্ণনার পূর্বে import স্টেটমেন্টের সাহায্যে সংশ্লিষ্ট ক্লাস বর্ণনাকারী প্যাকেজ ফাইল সংযুক্ত করতে হয়।

মেথড কলিং (Method calling) ঃ জাভা ভাষায় মেথড কলিং (Method calling) বলতে সাধারণত ক্লাসের মধ্যে পূর্বে ডিফাইনকৃত কোন মেথডকে main ( ) মেথডের অভ্যন্তরে অবজেক্ট সহযোগে ডিক্লেয়ার করা বুঝায়।

যখন একটি মেথড অপর কোন মেথডকে ব্যবহার করে, তখন তাকে ব্যবহারকারী বা ইউজার বা কলার মেথড এবং ব্যবহৃত মেথডকে কল্ড মেথড বলা হয়। মেথড কল প্রক্রিয়ায় একটি মেথডের সাথে অপর এক বা একাধিক মেথড সংযুক্ত করা হয়। মেথড কল একটি স্টেটমেন্ট। সুতরাং সাধারণত এর শেষে সেমিকোলন (;) থাকতে হবে। এক জোড়া প্রথম বন্ধনী '()' দেখে যেমন মেথড বুঝা যায়, তেমনি এক জোড়া প্রথম বন্ধনীর শেষে অতিরিক্ত একটি সেমিকোলন দেখে মেথড কল চেনা যায়। তবে মেথড কল কোন স্টেটমেন্টের মাঝে ব্যবহৃত হলে সেখানে কোন সেমিকোলন বসে না। জাভা ভাষায় দুই ভাগে মেথড কল করা যায়ঃ

- অপর কোন মেথডের মধ্যে এবং
- অবজেক্টের সাথে ডট অপারেটর ব্যবহার করে।

সিনটেক্স (Syntax) ঃ একটি মেথডের মধ্যে অপর কোন মেথড কল করার ফরম্যাট হলঃ

```
MethodName (Actual Argument List);
```

অবজেক্টের সাথে ডট অপারেটর ব্যবহার করে মেথড কল করার ফরম্যাট হল ঃ

```
ObjectName.MethodName(Actual Argument List);
```

উদাহরণ ঃ

```
student s = new student ( ); // object Declaration
s. Display ( ); // Method calling
```

অথবা

```
add test = new ( ); // object declaration
System. out. println (test. add (40, 42)),// method calling.
```

মেথডের রিটার্ন স্টেটমেন্ট (Method's return statement) ঃ প্রতিটি মেথডই কলিং মেথডে একটি বৈধ ডাটা টাইপের মান ফেরৎ পাঠায়। মেথড কর্তৃক মান ফেরত পাঠানোর জন্য return স্টেটমেন্ট ব্যবহার করা হয়।

return স্টেটমেন্ট ব্যবহারের ফরম্যাট হল ঃ

```
return ReturnValue;
```

অথবা,

```
return (ReturnValue);
```

return স্টেটমেন্ট এর কাজ ঃ

- কোন মেথডে return স্টেটমেন্ট execute হওয়ার সাথে সাথে ঐ মেথডের execution বন্ধ হয়ে যায় এবং প্রোগ্রামের control প্রতিস্থাপিত হয় calling মেথডে। সুতরাং, return ফাংশন থেকে exit হবার জন্য ব্যবহৃত হয়।
- return-এর সাথে কোন expression থাকলে সেই expression-এর মান calling মেথডে পাঠানোর জন্য return ব্যবহৃত হয়।
- কোন মেথড-এ return এরপর কোন statement থাকলে তা execute হবে না। যেমন-

```
if (i = = 0)
return (i);
++ i;
```

উপরের statement গুলিতে। যদি '0' হয় তবে ++i স্টেটমেন্ট execute হবে না এবং এই ফাংশন terminate (শেষ) হবে।

return স্টেটমেন্ট লেখার নিয়ম ঃ

- return এর সাথে কোন expression ব্যবহার না করলে তা মেথডের closing brace (}) হিসেবে কাজ করে। যেমন-

```
return; // এখানেই ফাংশন শেষ হবে।
```

- return-এর expression-এর মধ্যে ভেরিয়েবল বা value, উভয়ই ব্যবহার করা যায়। যেমন-

```
return (i);
return (100);
```

- return-এর expression-এ operator ব্যবহার করা যায়, যেমন-

```
return (x*y); // x ও y -এর মান গুণ হয়ে ফলাফল return হবে।
return (x+y);
return ++x;
```

ক্লাসে মেথড সংযোজনের প্রক্রিয়াটি একটি প্রোগ্রামের মাধ্যমে দেখানো হলো।

```
class Student
{
float Mark1, Mark2, Tutorial;
float Total,GTotal;
float GetMark(float M1,float M2)
{
Mark1=M1;
Mark2=M2;
Total=Mark1+Mark2;
return(Total);
}
void Display()
{
Tutorial=10;
GTotal=Total+GetMark(35,60);
System.out.println("Tutorail Mark is:"+Tutorial);
System.out.println("Mark1 is : \t"+Mark1);
System.out.println("Mark2 is : \t"+Mark2);
System.out.println("Grand Total is: " +GTotal);
}
}
public class ret
{
public static void main(String args[])
{
Student s=new Student();
s.Display();
}
}
```

৭.৫ মেথড ব্যবহারের সুবিধাসমূহ (Advantages of Method) ঃ

মেথড ব্যবহার করার ফলে অনেক সুবিধা পাওয়া যায়। যেমন-

- মেথড (Method) ছাড়া ক্লাসের ডাটা সরাসরি কেউ একসেস (Access) করতে পারে না। তাতে ডাটা Corruption এর সম্ভাবনা খুবই কমে যায়।
- মেথড ব্যবহার করার ফলে প্রোগ্রামের কোন অংশ মূল প্রোগ্রামে বার বার লিখার প্রয়োজন হয় না।
- প্রোগ্রামের সাইজ ছোট হয়।
- প্রোগ্রাম লিখতে সময় অপেক্ষাকৃত কম লাগে।
- প্রোগ্রাম পড়ে বুঝা ও সংশোধন করা সহজ হয়।
- বড় প্রোগ্রাম অনেকগুলো মডিউলে ভাগ করে নিয়ে এক এক মডিউল আলাদাভাবে বিভিন্ন প্রোগ্রামার দিয়ে প্রোগ্রাম করা হলে এবং এগুলো মেথড হিসাবে ব্যবহার করা হলে স্বল্পতম সময়ে প্রোগ্রাম সম্পন্ন করা সম্ভব হয়।
- অত্যন্ত জরুরি প্রোগ্রামকে মেথড হিসাবে লাইব্রেরিতে জমা করে রাখলে অন্য ব্যবহারকারী প্রয়োজনে তা ব্যবহার করতে পারে।

৭.৬ ওভারলোডিং মেথড (Overloading Method in Java) ঃ

মেথড ওভারলোডিং অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর একটি গুরুত্বপূর্ণ বৈশিষ্ট্য। জাভা ল্যাংগুয়েজে যদি একই ক্লাসের মধ্যে আরগুমেন্ট টাইপ কিংবা আরগুমেন্ট সংখ্যার ভিন্নতায় একই নামে দুই বা ততোধিক মেথড ব্যবহার করা হয় তবে তাকে মেথড ওভারলোডিং (Method overloading) বলে। আর এ প্রক্রিয়ায় যেসব মেথড ব্যবহৃত হয় তাদেরকে ওভারলডেড মেথড (Overloadded method) বলে। মেথড ওভারলোডিং এর বেলায় মেথডগুলোর আরগুমেন্ট টাইপ কিংবা আরগুমেন্ট সংখ্যার ভিত্তিতে কম্পাইলার সঠিক মেথডটি নির্বাচন করে। উল্লেখ্য যে, মেথড ওভারলোডিং এর বেলায় রিটার্ন টাইপের কোন ভূমিকা নেই।

উদাহরণ ঃ

```
int cal (int x, int y);
float cal (float m1, float m2);   // Overloaded method
int cal (int x);
```

বিভিন্ন টাইপ আরগুমেন্ট ব্যবহার করে মেথড ওভারলোডিং-এর একটি প্রোগ্রাম দেয়া হলঃ

```
class over
{
int cal(int x,int y)
{
int z;
z=x+y;
System.out.println("Using summation Method");
System.out.println("Result is :"+z);
return 0;
}
float cal(float m1, float m2)
```

```
{
float mark;
mark=m1+m2;
System.out.println("Using Float Method");
System.out.println("The Mark is:"+mark);
return 0;
}
int cal(int x)
{
System.out.println("Using Input/Output Method");
System.out.println(x);
return 0;
}
}
public class overloading
{
public static void main(String args[])
{
over s=new over();
s.cal(37.25f, 50.26f);
s.cal(44);
s.cal(23,10);
}
}
```

মেথড ওভারলোডিং-এর সীমাবদ্ধতা (Limitation's of method overloading) ঃ মেথড ওভারলোডিং প্রক্রিয়ার প্রধান সীমাবদ্ধতা হল মেথড ওভারলোডিং প্রক্রিয়ায় ব্যবহৃত মেথডগুলোর ক্ষেত্রে রিটার্ন টাইপের কোন ভূমিকা নেই। আর একই নামে ভিন্ন ভিন্ন আরগুমেন্ট টাইপ কিংবা আরগুমেন্ট সংখ্যার একাধিক মেথড ব্যবহার করা গেলেও প্রতিটি মেথড আলাদাভাবে বর্ণনা করতে হয়। অর্থাৎ মেথড ওভারলোডিং প্রক্রিয়ায় একই গ্রুপের একাধিক মেথড একটি কমন বর্ণনার মাধ্যমে ব্যবহার করা যায় না।

৭.৭ কনস্ট্রাক্টর ও ওভারলোডিং কনস্ট্রাক্টর (Constructor and Overloading Constructor in Java) ঃ

কনস্ট্রাক্টর (Constructor) ঃ কনস্ট্রাক্টর (Constructor) এমন এক বিশেষ ধরনের মেথড যা কোন অবজেক্ট ঘোষণার সময় তার প্রারম্ভিক মান নির্ধারণ এবং মেমোরি স্পেস বরাদ্দের জন্য ব্যবহৃত হয়। কনস্ট্রাক্টর খুবই গুরুত্বপূর্ণ, কারণ কনস্ট্রাক্টরের মাধ্যমেই অবজেক্টের প্রপার্টি বা ফিল্ড (Field) গুলোর প্রারম্ভিক মান (initial value) প্রদান করে। তাছাড়া ক্লাসে কোন কনস্ট্রাক্টর না থাকলেও জাভা কম্পাইলার একটি deafult কনস্ট্রাক্টর প্রদান করে যা ফিল্ড ভেরিয়েবলগুলোর ডিফল্ট ভ্যালু (Default value) নিয়ে একটি অবজেক্ট তৈরি করে।

কনস্ট্রাক্টর মেথডের বৈশিষ্ট্য (Features of constructor method) ঃ

- কনস্ট্রাক্টর মেথডের নাম তার ধারণকারী ক্লাসের নামের অনুরূপ হয়। অর্থাৎ ক্লাস ও মেথডের নাম একই (same)।
- কনস্ট্রাক্টর মেথড সবসময় ক্লাসের পাবলিক অংশে ঘোষণা করা হয়।
- কনস্ট্রাক্টর মেথড কোন মান রিটার্ন করতে পারে না। কারণ এর কোন রিটার্ন টাইপ নেই, এমন কি void টাইপ ও নয়।
- কনস্ট্রাক্টর মেথড কল না করলেও হয়। কোন ক্লাসের অবজেক্ট ঘোষণার সময় কম্পাইলার স্বয়ংক্রিয়ভাবে সেই ক্লাসের কনস্ট্রাক্টর মেথড আহ্বান করে।
- কনস্ট্রাক্টর মেথড ইনহেরিট করা যায় না। যদিও কোন ডিরাইভড ক্লাস কনস্ট্রাক্টর বেস ক্লাস কনস্ট্রাক্টর কল করতে পারে।
- কনস্ট্রাক্টর মেথড আরগুমেন্ট বিশিষ্ট হতে পারে।

কনস্ট্রাক্টর মেথডের ডিক্লারেশন (Constructor declaration) ঃ কনস্ট্রাক্টর মেথড ডিক্লারেশনের ফরম্যাট হল ঃ

```
class ClassName
{
private members;
protected members;
public ClassName ( )
{
// Body
}
}
```

উদাহরণ ঃ

```
class Student
{
long Roll;
String Name;
float Mark;
public Student()              // Constructor Method
{
//.....
}
void Display()                // Normal Method
{
//.....
}
}
}
```

উল্লেখ্য, কনস্ট্রাক্টর মেথড কল করতে হয় না, কনস্ট্রাক্টর মেথড বিশিষ্ট ক্লাসের অবজেক্ট ঘোষণার সময় কম্পাইলার স্বয়ংক্রিয়ভাবে কনস্ট্রাক্টর মেথড কল করে।

নিম্নে কনস্ট্রাক্টর মেথড ব্যবহার করে একটি প্রোগ্রাম দেয়া হল ঃ

```
class Student
{
int Roll;
String Name;
float Mark;
public Student() //Constructor Method
{
Roll=96004;
Name="Munny";
Mark=84.75f;
System.out.println("Roll is :"+Roll);
System.out.println("Name is :"+Name);
System.out.println("Mark is :"+Mark);
}
}
public class cons
{
public static void main(String args[])
{
Student s=new Student();
}
}
```

ওভারলোডিং কনস্ট্রাক্টর (Overloading constructor) ঃ কোন কনস্ট্রাক্টর মেথডের আরগুমেন্ট হিসেবে একই ক্লাসের কোন অবজেক্ট ব্যবহৃত হলে তাকে কপি কনস্ট্রাক্টর বলা হয়। কপি কনস্ট্রাক্টরের ক্ষেত্রে একই নামে একাধিক কনস্ট্রাক্টর ফাংশন ব্যবহৃত হয় বলে একে ওভারলোডিং কনস্ট্রাক্টরও বলা হয়। নিম্নে কপি কনস্ট্রাক্টরের গঠন ও তার ব্যবহার সম্পর্কিত একটি প্রোগ্রাম দেয়া হল ঃ

```
class Student
{
int Roll;
String Name;
float Mark;
Student(int R, String N, float M) //Argumented Constructor Method
```

```
{
Roll=R;
Name=N;
Mark=M;
}
Student(Student s)        //Copy Constructor
{
Roll=s.Roll;        //Copy Constructor Define
Name=s.Name;
Mark=s.Mark;
}
void Display()
{
System.out.println("Roll is :"+Roll);
System.out.println("Name is :"+Name);
System.out.println("Mark is :"+Mark);
}
}

public class copy
{
public static void main(String args[])
{
Student s1=new Student(101,"Tonny",65.23f);
System.out.println("Record of S1");
s1.Display();
System.out.println();
Student s2=new Student(s1);  //Copy Constructor
System.out.println("Record of S2");
s2.Display();
System.out.println("Both are same...");
}
}
```

৭.৮ ইন্সটেন্স ভেরিয়েবল হাইডিং ও গারবেজ কালেকশন (Instance Variable Hiding and Garbage Collection) ঃ

ইন্সটেন্স ভেরিয়েবল হাইডিং (Instance variable hiding) ঃ কনস্ট্রাক্টর মেথডের মাধ্যমে কোন ক্লাসের অবজেক্টের ফিল্ড বা ইন্সটেন্স ভেরিয়েবলগুলো ইনশিয়ালাইজ (initialize) হয় এবং এভাবে একটি ক্লাসের অবজেক্ট তৈরি হয়। ক্লাসে যখন কোন অবজেক্ট তৈরি হয় তখন তা মেমোরিতে স্পেস দখল করে। কিন্তু পরবর্তীতে ঐ অবজেক্টটির এক্সিকিউশন (Execution) বন্ধ হয়ে গেলেও ঐ অবজেক্ট কর্তৃক দখলকৃত মেমোরি খালি হয় না। ফলে পরবর্তীতে মেমোরি shortage দেখা দেয়। তাই জাভা ভাষায় এমন এক ধরনের মেকানিজম রয়েছে যা ব্যবহার করে ইন্সটেন্স ভেরিয়েবলকে হাইড (Hide) করা যায় এবং মেমোরির অপচয়রোধ করা যায়। সাধারণত finalize ( ) মেথড ব্যবহার করে ইন্সটেন্স ভেরিয়েবল হাইডিং প্রক্রিয়া সম্পন্ন করা যায়।

গারবেজ কালেকশন (Garbage collection) ঃ গারবেজ কালেকশন এমন একটি স্বয়ংক্রিয় পদ্ধতি যা প্রোগ্রামে অব্যবহৃত অবজেক্ট কিংবা দীর্ঘক্ষণ যাবৎ ব্যবহৃত হচ্ছে না এমন সব অবজেক্ট কর্তৃক বরাদ্দকৃত মেমোরি স্পেস খালি করে দেয়।

একটি উদাহরণের মাধ্যমে গাববেজ কালেকশনের ব্যাপারটি আরো ভালভাবে পরিষ্কার করা যাক, একটি স্ট্রিং ভেরিয়েবলে "১০:১৫ মিনিট" সময় সংরক্ষিত আছে, ১ মিনিট পর ঐ ভেরিয়েবলের নতুন মান হবে "১০:১৬ মিনিট "। এখন আগের মেমোরির অর্থাৎ "১০:১৫ মিনিট" স্টোর করার সেলটা কোন ভেরিয়েবল ব্যবহার করছে না, অর্থাৎ অন্য কোন ভেরিয়েবল এই স্ট্রিংকে ব্যবহার না করে তাহলে এটিই হবে গারবেজ। এই গারবেজ কে কালেক্ট করে মেমোরি খালি করার নামই গারবেজ কালেকশন।

বিভিন্নভাবে গারবেজ কালেকশন হয়ে থাকে। যেমন finalize ( ) মেথড ও system ক্লাসের gc ( ) মেথড কল করার মাধ্যমে গারবেজ কালেকশন করা হয়।

৭.৯ ক্লাস, অবজেক্ট, মেথড ও কনস্ট্রাক্টর ব্যবহার করে প্রোগ্রাম (Java Programs Relating to Class, Object, Method and Constructor) ঃ

উক্ত অধ্যায়ের পূর্বের টপিক্সগুলোতে এবং ব্যবহারিক অংশে এ সংক্রান্ত প্রোগ্রাম দেয়া আছে।

অনুশীলনী-৭

অতি সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। ক্লাস ডিক্লারেশন বলতে কী বুঝায়?

উত্তরঃ জাভা ভাষায় ক্লাস ডিক্লারেশন বলতে ক্লাসের মেম্বার ভেরিয়েবল এবং মেম্বার ফাংশনগুলোর ডিক্লারেশন ও ডেফিনিশন বুঝায়। ক্লাস ডিক্লারেশনের জন্য class কীওয়ার্ডের পরে ঐ ক্লাসের নাম দিতে হয়, ক্লাস বডি দ্বিতীয় বন্ধনী '{ }' দ্বারা সীমাবদ্ধ থাকে। উল্লেখ্য, সি++ এর মত জাভা ক্লাস সেমিকোলন (;) দিয়ে শেষ হয় না। ক্লাস বডিতে ক্লাসের মেম্বার ভেরিয়েবল এবং মেম্বার ফাংশনগুলো তাদের এক্সেসিবিলিটি মোডসহ উল্লেখ করা হয়।

২। জাভায় ক্লাস ডিক্লারেশনের ফরম্যাট লেখ।

উত্তরঃ জাভায় ক্লাস ডিক্লারেশনের ফরম্যাট নিম্নরূপ-

```
class ClassName
{
Member Variable Declaration;
Member Function Definition;
}
```

৩। জাভায় ক্লাসে অবজেক্ট ডিক্লারেশনের ফরম্যাট রেখ।

উত্তরঃ ক্লাসে অবজেক্ট ডিক্লারেশনের ফরম্যাট নিম্নরূপ ঃ

```
ClassName ObjectName;
ObjectName = new ClassName ( );
```

৪। ক্লাস মেম্বার এক্সেসিবিলিটি মোডগুলোর নাম লেখ।

উত্তরঃ ক্লাস মেম্বার এক্সেস করার জন্য তিন ধরনের এক্সেসিবিলিটি মোড বা ভিজিবিলিটি (Visibility) মোড রয়েছে। যথা ঃ

- এক্সেসিং পাবলিক মেম্বার (Accessing public member)
- এক্সেসিং প্রাইভেট মেম্বার (Accessing private members)
- এক্সেসিং প্রটেকটেড মেম্বার (Accessing protected member)

৫। মেথড কাকে বলে?

উত্তরঃ মেথড (method) হচ্ছে একটি বড় প্রোগ্রামের একটি স্বাধীন ক্ষুদ্র অংশ (module) যার একটি নির্দিষ্ট নাম থাকে, যা এক বা একাধিক স্টেটমেন্টের সমন্বয়ে গঠিত এবং নির্দিষ্ট কোন সমস্যা সমাধানের জন্য ব্যবহৃত হয়। কোন ক্লাস মেথডের মাধ্যমেই অবজেক্টকে কোন কার্যসম্পাদনের ক্ষমতা প্রদান করে।

৬। মেথডে ব্যবহৃত একসেস মডিফায়ারগুলোর নাম লেখ।

উত্তরঃ মেথডে ব্যবহৃত একসেস মডিফায়ারগুলো হল ঃ

- প্রাইভেট (Private)
- পাবলিক (Public) ও
- প্রটেকটেড (Protected) ইত্যাদি।

৭। **মেথডের উপাদানগুলো উল্লেখ কর।**

উত্তরঃ উপাদানগুলো হল ঃ

- মেথড ডেফিনিশন (Method definition)
- প্রটোটাইপ (Prototype)
- মেথড কলিং (Method calling)
- মেথডের রিটার্ন স্টেটমেন্ট (Methods return statement)

৮। **মেথডের প্রটোটাইপ বলতে কী বুঝায়?**

উত্তরঃ মেথডের প্রটোটাইপ বলতে তার সঠিক নাম, রিটার্ন টাইপ, আরগুমেন্ট তালিকা, সুনির্দিষ্ট ফরম্যাটে তার ব্যবহারবিধি ইত্যাদি বুঝায়।

৯। **মেথড ওভারলোডিং কাকে বলে?**

উত্তরঃ জাভা ল্যাংগুয়েজে যদি একই ক্লাসের মধ্যে আরগুমেন্ট টাইপ কিংবা আরগুমেন্ট সংখ্যার ভিন্নতায় একই নামে দুই বা ততোধিক মেথড ব্যবহার করা হয় তবে তাকে মেথড ওভারলোডিং (Method overloading) বলে।

১০। **কনস্ট্রাক্টর কী?**

উত্তরঃ কনস্ট্রাক্টর (Constructor) এমন এক বিশেষ ধরনের মেথড যা কোন অবজেক্ট ঘোষণার সময় তার প্রারম্ভিক মান নির্ধারণ এবং মেমোরি স্পেস বরাদ্দের জন্য ব্যবহৃত হয়।

১১। **ওভারলোডিং কনস্ট্রাক্টর বা কপি কনস্ট্রাক্টর কাকে বলে?**

উত্তরঃ কোন কনস্ট্রাক্টর মেথডের আরগুমেন্ট হিসেবে একই ক্লাসের কোন অবজেক্ট ব্যবহৃত হলে তাকে কপি কনস্ট্রাক্টর বলা হয়। কপি কনস্ট্রাক্টরের ক্ষেত্রে একই নামে একাধিক কনস্ট্রাক্টর ফাংশন ব্যবহৃত হয় বলে একে ওভারলোডিং কনস্ট্রাক্টরও বলা হয়।

১২। **গারবেজ কালেকশন কী?**

উত্তরঃ গারবেজ কালেকশন এমন একটি স্বয়ংক্রিয় পদ্ধতি যা প্রোগ্রামে অব্যবহৃত অবজেক্ট কিংবা দীর্ঘক্ষণ যাবৎ ব্যবহৃত হচ্ছে না এমন সব অবজেক্ট কর্তৃক বরাদ্দকৃত মেমোরি স্পেস খালি করে দেয়।

১৩। **কনস্ট্রাক্টর কত প্রকার ও কী কী?**

উত্তরঃ কনস্ট্রাক্টর তিন প্রকার। যথা ঃ

- ডিফল্ট বা আরগুমেন্টবিহীন কনস্ট্রাক্টর
- আরগুমেন্টেড কনস্ট্রাক্টর
- কপি কনস্ট্রাক্টর

১৪। **মাল্টিপল কনস্ট্রাক্টর কী?**

উত্তরঃ কোন ক্লাসে একই সাথে দুই বা ততোধিক আরগুমেন্টেড কনস্ট্রাক্টর মেথড থাকলে তাকে মাল্টিপল কনস্ট্রাক্টর বলে।

সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। **জাভায় ক্লাস ডিক্লারেশনের ফরম্যাট উদাহরণসহ উল্লেখ কর।**

উত্তর সংকেত ঃ ৭.১ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **অবজেক্ট রেফারেন্স ভেরিয়েবল এসাইন করার প্রক্রিয়া বর্ণনা কর।**

উত্তর সংকেত ঃ ৭.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। জাভা মেথডের বিভিন্ন অংশ বর্ণনা কর।

উত্তর সংকেত ঃ ৭.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। রিটার্ন স্টেটমেন্টের কাজ লেখ।

উত্তর সংকেত ঃ ৭.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। রিটার্ন স্টেটমেন্ট লেখার নিয়মাবলি কী কী?

উত্তর সংকেত ঃ ৭.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। মেথড ব্যবহারের সুবিধাসমূহ উল্লেখ কর।

উত্তর সংকেত ঃ ৭.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৭। মেথড ওভারলোডিং এর সীমাবদ্ধতা কী কী?

উত্তর সংকেত ঃ ৭.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

৮। কনস্ট্রাক্টর মেথডের বৈশিষ্ট্যগুলো উল্লেখ কর।

উত্তর সংকেত ঃ ৭.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

৯। ইন্সটেন্স ভেরিয়েবল হাইডিং প্রক্রিয়া বর্ণনা কর।

উত্তর সংকেত ঃ ৭.৮ নং অনুচ্ছেদ দ্রষ্টব্য।

১০। উদাহরণসহ গারবেজ কালেকশন পদ্ধতি ব্যাখ্যা কর।

উত্তর সংকেত ঃ ৭.৮ নং অনুচ্ছেদ দ্রষ্টব্য।

➥ *রচনামূলক প্রশ্নোত্তর ঃ*

১। ক্লাসের পাবলিক মেম্বার এক্সেসিং প্রক্রিয়া প্রোগ্রাম লিখে ব্যাখ্যা কর।

উত্তর সংকেত ঃ ৭.১.১ নং অনুচ্ছেদ দ্রষ্টব্য।

২। একটি প্রোগ্রামের মাধ্যমে ক্লাসের প্রাইভেট মেম্বার এক্সেসিং প্রক্রিয়া বর্ণনা কর।

উত্তর সংকেত ঃ ৭.১.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। ক্লাসের প্রটেকটেড মেম্বার এক্সেস করার প্রক্রিয়া প্রোগ্রাম লিখে বর্ণনা কর।

উত্তর সংকেত ঃ ৭.১.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। ক্লাসে মেথড সংযোজনের প্রক্রিয়াটি প্রোগ্রামের মাধ্যমে দেখাও।

উত্তর সংকেত ঃ ৭.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। মেথড ওভারলোডিং পদ্ধতির জন্য একটি জাভা প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ৭.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। কনস্ট্রাক্টর মেথড ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ৭.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

৭। প্রোগ্রামের মাধ্যমে ওভারলোডিং কনস্ট্রাক্টর বা কপি কনস্ট্রাক্টর পদ্ধতি ব্যাখ্যা কর।

উত্তর সংকেত ঃ ৭.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

অধ্যায় ০৮

ইনহেরিটেন্স ও পলিমরফিজম
Inheritance and Polymorphism

৮.০ ভূমিকা (Introduction) ঃ

পুনঃব্যবহারযোগ্যতা (Reusability) অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং-এর একটি গুরুত্বপূর্ণ বৈশিষ্ট্য। একই প্রোগ্রামে কিছু অংশ বারবার লেখার চেয়ে একবার লিখে তা পুনঃব্যবহার করা সুবিধাজনক। এতে শুধু সময় ও খরচ কমে না, বরং ঝামেলা কমে এবং প্রোগ্রামের সৌন্দর্য ও গ্রহণযোগ্যতা বৃদ্ধি পায়। পূর্বে তৈরি কোন ক্লাস যা নির্ভুল, পরীক্ষিত এবং এক বা একাধিকবার ব্যবহৃত, তা নতুন করে তৈরি করার চেয়ে পুনরায় ব্যবহার করা নিশ্চয়ই সুবিধাজনক। পূর্ণাঙ্গ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং হিসেবে জাভা পুরোপুরি এই সুবিধা দেয়। জাভা ভাষায় ইনহেরিটেন্স ও ইন্টারফেস প্রক্রিয়ার মাধ্যমে ক্লাসের পুনঃব্যবহার প্রক্রিয়া বাস্তবায়ন করা হয়। জাভা ভাষায় একই ক্লাস বিভিন্নভাবে পুনঃব্যবহার করা যায়। যদি কোন প্রোগ্রামার তার প্রোগ্রামের জন্য কোন ক্লাস ডিফাইন করে তা সংরক্ষণ করেন তবে অন্যান্য প্রোগ্রামারগণও তাদের প্রোগ্রামে এই ক্লাস ব্যবহার করতে পারেন। পুরাতন কোন ক্লাস থেকে নতুন ক্লাস তৈরি করার কৌশলকে ইনহেরিটেন্স বলা হয়। ইনহেরিটেন্স প্রক্রিয়ায় পুরাতন এক বা একাধিক ক্লাস থেকে নতুন এক বা একাধিক ক্লাস তৈরি করা হয়। সে ক্ষেত্রে পুরাতন ক্লাস বা ক্লাসসমূহ বেস ক্লাস এবং নতুন ক্লাস বা ক্লাসসমূহকে ডিরাইভড ক্লাস বা ইনহেরিটেড ক্লাস বলা হয়। তবে জাভা ভাষায় সি++ এর মত মাল্টিপল ইনহেরিটেন্স প্রক্রিয়া অর্থাৎ কোন ডিরাইভড ক্লাসের জন্য দুই বা ততোধিক বেস ক্লাস ব্যবহার অনুমোদন করে না। অন্যদিকে পলিমরফিজম হচ্ছে ভিন্ন ভিন্ন অবস্থায় একই অবজেক্টের ভিন্ন ভিন্ন ফল। এই প্রক্রিয়ায় ভিন্ন ভিন্ন ক্লাসের কতকগুলো অবজেক্টকে একই গ্রুপে বর্ণনা করা হয় এবং কম্পাইলার অবজেক্টগুলোর রেফারেন্স টাইপের উপর ভিত্তি করে উপযুক্ত ফাংশন কল করে স্ব স্ব কার্য নির্বাহ করে।

৮.১ সুপার ক্লাস ও সাব ক্লাস (Super Class and sub Class) ঃ

ইনহেরিটেন্স (Inheritance) ঃ যে প্রক্রিয়ায় পূর্বে ডিফাইনকৃত এক বা একাধিক ক্লাস হতে গুণাগুণ ও বৈশিষ্ট্য ইনহেরিট করে নতুন কোন ক্লাস তৈরি করা হয় তাকে ইনহেরিটেন্স (Inheritance) বলে। ইনহেরিটেন্স প্রক্রিয়ার মাধ্যমে ক্লাসের পুনঃব্যবহার (reusability) প্রক্রিয়া বাস্তবায়ন করা হয়। এতে করে প্রোগ্রামের সময় খরচ ও ঝামেলা কমার পাশাপাশি প্রোগ্রামের সৌন্দর্য ও গ্রহণযোগ্যতা বৃদ্ধি পায়।

সুপার ক্লাস (Super class) ঃ ইনহেরিটেন্স প্রক্রিয়ায় পূর্বে ডিফাইনকৃত যে এক বা একাধিক ক্লাস হতে বৈশিষ্ট্য ইনহেরিট করা হয় সে বা সে সকল ক্লাসকে সুপার ক্লাস (Super class) বলে।

সাব ক্লাস (Sub class) ঃ ইনহেরিটেন্স প্রক্রিয়ায় পূর্বে ডিফাইনকৃত এক বা একাধিক ক্লাস হতে বৈশিষ্ট্য ইনহেরিট করে নতুন যে ক্লাস তৈরি হয় সেই ক্লাসকে সাব ক্লাস (sub class) বলে।

উদাহরণ ঃ

```
Class result extends student
      ↓                ↓
  sub class        super class
```

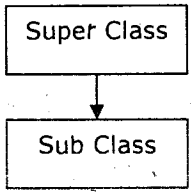
উল্লেখ্য যে, সাব ক্লাস, সুপার ক্লাসের protected ও public member গুলোই কেবল ইনহেরিট করতে পারে কিন্তু private member গুলো কখনোই ইনহেরিট করতে পারে না।

৮.২ বিভিন্ন প্রকার ইনহেরিটেন্স (Single Inheritance, Multi Level Inheritance and Hierarchical Inheritance) ঃ

সুপার ক্লাস ও সাব ক্লাসের সংখ্যা এবং সাব ক্লাস গঠনের কৌশলের ভিত্তিতে জাভা ভাষায় ইনহেরিটেন্স প্রক্রিয়াকে মূলত তিন ভাগে ভাগ করা হয় ঃ

- সিঙ্গেল ইনহেরিটেন্স (Single Inheritance)।
- মাল্টিলেভেল ইনহেরিটেন্স (Multilevel Inheritance)।
- হায়ারারকিক্যাল ইনহেরিটেন্স (Hierarchical Inheritance)।

সিঙ্গেল ইনহেরিটেন্স (Single Inheritance) ঃ যখন কোন সাব ক্লাস কেবল একটিমাত্র বেস ক্লাস থেকে এক বা একাধিক বৈশিষ্ট্য ইনহেরিট করে, তখন সেই প্রক্রিয়াকে সিঙ্গেল ইনহেরিটেন্স (Single Inheritance) বলা হয়। এরূপ সুপার ক্লাসকে ডাইরেক্ট সুপার ক্লাস বলা হয়। সিঙ্গেল ইনহেরিটেন্স-এর ব্লক চিত্র এবং ফরম্যাট দেয়া হলঃ



চিত্র ঃ সিঙ্গেল ইনহেরিটেন্স

সিনটেক্স (Syntax) ঃ

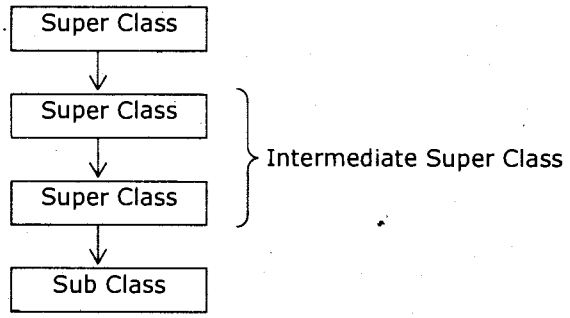
```
class A
{
: // class body
}
class B extends A
{
M / / class body
}
```

সিঙ্গেল ইনহেরিটেন্স প্রক্রিয়া বাস্তবায়নের একটি নমুনা প্রোগ্রাম দেয়া হলঃ

```
class Student
{
int Roll;
String Name=new String();
}
class Result extends Student    //Inheriting Student Class
{
float Mark;
void GetData()
{
Roll=96006;
Name="Sumaiya";
Mark=70.75f;
}
```

```
void Display()
{
System.out.println(" Roll is :"+Roll);
System.out.println(" Name is :"+Name);
System.out.println(" Mark is :"+Mark);
}
}
class inherit
{
public static void main(String args[])
{
Result R=new Result();
R.GetData();
R.Display();
}
}
```

মাল্টিলেভেল ইনহেরিটেন্স (Multilevel Inheritance) ঃ যখন কোন সাব ক্লাস অপর কোন সুপার ক্লাসের সাব ক্লাস থেকে এক বা একাধিক বৈশিষ্ট্য ইনহেরিট করে, তখন সেই প্রক্রিয়াকে মাল্টিলেভেল ইনহেরিটেন্স (Multilevel Inheritance) বলা হয়। মাল্টিলেভেল ইনহেরিটেন্স প্রক্রিয়ায় ব্লক- চিত্র এবং ফরম্যাট দেয়া হল ঃ



চিত্র ঃ মাল্টিলেভেল ইনহেরিটেন্স

সিনটেক্স (Syntax) ঃ

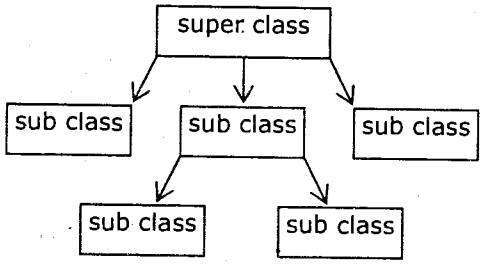
```
class A
  {
    ¦        / / Class body
  }
 class B extends A
  {
    ¦          / / class body
  {
class C extends B
  }
    ¦          \ \ class body
  }
```

মাল্টিলেভেল ইনহেরিটেন্স প্রক্রিয়া ব্যবহার করে নিম্নে একটি প্রোগ্রাম দেয়া হল ঃ

```
class Student
{
int Roll;
String Name=new String();
void GetData()
{
Roll=96002;
Name="Tahsin";
}
void Display()
{
System.out.println(" Inside Student Class");
System.out.println(" Roll is :"+Roll);
System.out.println(" Name is :"+Name);
}
}
class Exam extends Student     //Inheriting Student Class
{
float Mark;
void GetData()
{
Mark=80.75f;
}
void Display()
{
System.out.println(" Inside Exam Class");
System.out.println(" Mark is  :"+Mark);
}
}
class Result extends Exam
{
Student S=new Student();
Exam E=new Exam();
void GetData()
{
S.GetData();
E.GetData();
}
void Display()
{
```

```
S.Display();
E.Display();
}
}
class multi
{
public static void main(String args[])
{
Result R=new Result();
R.GetData();
R.Display();
}
}
```

হায়ারারকিক্যাল ইনহেরিটেন্স (Hierarchical Inheritance) ঃ যখন একটি মাত্র সুপার ক্লাস হতে দুই বা ততোধিক সাব ক্লাস গঠন করা হয়, তখন সেই প্রক্রিয়াকে হায়ারারকিক্যাল ইনহেরিটেন্স (Hierarchical Inheritance) বলা হয়। নিম্নে হায়ারারকিক্যাল ইনহেরিটেন্স-এর ব্লক চিত্র ও ফরম্যাট দেয়া হল ঃ



চিত্র ঃ হায়ারারকিক্যাল ইনহেরিটেন্স

সিনটেক্স (Syntax) ঃ

```
class A
{
//... .... ...
}
class B extends A
{
//.....
}
class C extends A
{
//.....
}
class D extends C
{
```

```
//.....
}
class E extends C
{
//.....
}
class N extends A
{
//.....
}
```

নিম্নে হায়ারারকিক্যাল ইনহেরিটেন্স প্রক্রিয়া বাস্তবায়নের একটি নমুনা প্রোগ্রাম দেয়া হল ঃ

```
class Student
{
int Roll;
String Name=new String();
void GetName()
{
Name="Tahsin";
System.out.println(" Name is :"+Name);
}
}
class Academic extends Student
{
int Roll;
Void GetRoll()
{
Roll=96002;
System.out.println(" Roll is :"+Roll);
}
}
class Result extends Student
{
float Mark;
void GetMark()
{
Mark=80.25f;
System.out.println("Mark is : +" Mark);
}
}
class hierarchical
{
```

```
public static void main(String args[])
{
Academic A=new Academic();
A.GetRoll();
Student S=new Student();
S.GetName();
Result R=new Result();
R.GetMark();
}
}
Result R=new Result();
R.GetData();
R.Display();
}
}
```

৮.৩ জাভা ভাষায় ওভাররিডেন মেথড (Overridden Method in Java) ঃ

কোন মেথড যদি সুপার ক্লাস এবং সাব-ক্লাস দুটিতেই বিদ্যমান থাকে তবে সুপার ক্লাস-এর মেথডটিকে ওভাররিডেন মেথড (Overridden method) বলে। আর এর করেসপন্ডিং (Correspondin) সাব-ক্লাস-এর মেথডটিকে ওভাররাইডিং মেথড (Overriding method) বলে। এক্ষেত্রে অবজেক্ট যখন মেথড কল করে তখন শুধুমাত্র রেসপেক্টিভ ক্লাসের মেথডই invoke হয় , তার সুপার অথবা সাব ক্লাস invoke হয় না। সুপার ক্লাসকে ইনহেরিট করে সাব-ক্লাস তৈরি করা হলে সুপার ক্লাসের সকল পাবলিক ও প্রটেকটেড মেথড সাব-ক্লাসে চলে আসে। কোন সুপার ক্লাস মেথড-এর implementation বা body সাব-ক্লাস-এর উপযোগী করার জন্য সুপার ক্লাস-এর মেথডকে সাব ক্লাস এ Override করা হয়।

নিম্নে Overridden Method ব্যবহার করে একটি প্রোগ্রাম দেয়া হল ঃ

```
class SuperClass
{
int n;
SuperClass()
{
n++;
}
public void GetData()
{
System.out.println("It is SuperClass");
System.out.println("n= "+n);
}
}
class SubClass extends SuperClass
{
```

```
int n;
SubClass()
{
n++;
}
public void GetData()
{
System.out.println("It is SubClass");
System.out.println("n= "+n);
}
}
public class overr
{
public static void main(String args[])
{
SuperClass sp=new SubClass();
sp.GetData();
SubClass sb=new SubClass();
sb.GetData();
}
}
```

ওভাররিডেন মেথডের বৈশিষ্ট্য (Features of Overridden method) ঃ

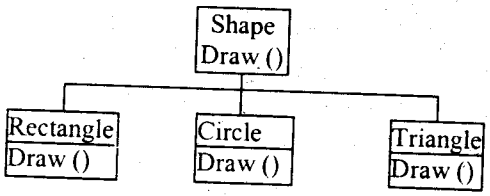
- কোনো একটি method এর নাম এবং argument-এর সংখ্যা, ধরন ও ক্রম যদি হুবহু parent বা superclass এর corresponding method এর মতো হয় তবে তা হল overriden method।
- প্রতিটি parent class এর method কে এর Subclass এ একবার মাত্রই override করা যাবে অর্থাৎ কোনো একটি class এ একই signature এর দুইটি method থাকবে না।
- Overriden method এর return type অবশ্যই অনুরূপ হতে হবে যাকে সে override করেছে।
- Overriden method এর access modifier কখনো more private হতে পারবে না। অর্থাৎ Superclass এর কোন method যদি public হয় তবে Subclass এ একে কখনো protected অথবা private বা friendly করা যাবে না।

৮.৪ জাভায় ডাইনামিক রান টাইম পলিমরফিজম (Dynamic Runtime Polymorphism in Java) ঃ

ডাইনামিক রান টাইম পলিমরফিজম বলতে 'পলিমরফিজম' কেই বোঝায়। দুটি গ্রিক শব্দ 'Poly' এবং 'morphism' মানে 'বহু' এবং 'রূপ' শব্দ যোগে 'Polymorphism' শব্দটি গঠিত হয়েছে। সুতরাং পলিমরফিজম মানে বহুরূপতা, অর্থাৎ ভিন্ন ভিন্ন অবস্থায় একই অবজেক্টের (মেথডের) ভিন্ন ভিন্ন ফল। অথবা বলা যায় একই মেথড ভিন্ন ভিন্ন ক্লাসে ভিন্ন ভিন্ন ভাবে কাজ করছে।

পলিমরফিজম প্রক্রিয়ায় কম্পাইলার প্রোগ্রাম চলাকালীন সময় সুপার ক্লাস ও সাব ক্লাসে একই প্রটোটাইপ বিশিষ্ট দুই বা ততোধিক মেম্বার মেথড এর মধ্য থেকে উপযুক্ত মেথড নির্বাচন করে। এ প্রক্রিয়ায় ভিন্ন ভিন্ন ক্লাসের কতগুলো অবজেক্ট কে একই গ্রুপে ডিফাইন করা হয় এবং কম্পাইলার অবজেক্টগুলোর রেফারেন্স টাইপের ভিত্তিতে উপযুক্ত মেথড কল করে স্ব স্ব কার্য নির্বাহ করে।

নিম্নে একটি উদাহরণের মাধ্যমে পলিমরফিজম বিষয়টি বোঝানোর চেষ্টা করা হল–



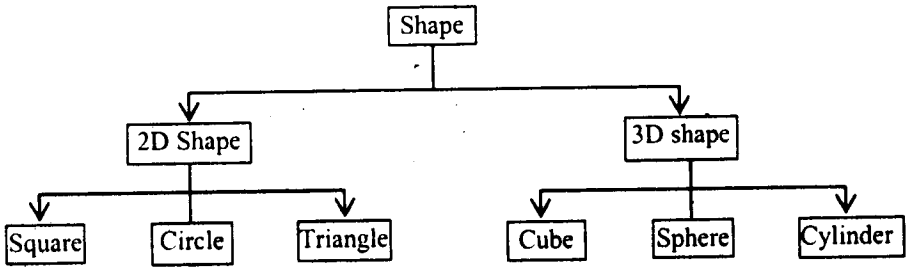
চিত্র ঃ পলিমরফিজম (Polymorphism)

উপরের চিত্রটি লক্ষ করলে দেখা যায় যে, 'shape' সুপার ক্লাসে 'Draw' নামে একটি মেথড আছে। অনুরূপভাবে, 'Rectangle', 'Circle' এর 'Triangle' এ তিনটি সাব ক্লাসে ও 'Draw' মেথডটি আছে। সংশ্লিষ্ট ক্লাসের ধরন অনুযায়ী প্রতিটি ক্লাসেই 'Draw' মেথডকে প্রয়োজন অনুযায়ী ভিন্ন ভিন্নভাবে ডিফাইন (Define) করা আছে। এমতাবস্থায় 'Rectangle' ক্লাসের অবজেক্টকে কল করলে Rectangle draw হবে, 'Circle' ক্লাসের 'Draw' অবজেক্ট কল করলে Circle draw হবে এবং Triangle ক্লাসের 'Draw' অবজেক্টকে কল করলে Triangle draw হবে। এটিই পলিমরফিজম।

উপরের কাজটি করার জন্য একটি অ্যারে (Array) বা ভেক্টর (Vector) তৈরি করে যতগুলো অবজেক্ট আছে সেগুলোকে অ্যারের এক একটি উপাদান (Element) হিসেবে ব্যবহার করা হয়। পলিমরফিজম প্রক্রিয়ার একটি মূল সুবিধা হচ্ছে এই যে, নতুন কোন অবজেক্ট 'ellipse draw' করতে চাইলে প্রোগ্রামে খুব বেশি পরিবর্তন করতে হবে না। শুধুমাত্র অ্যারে বা ভেক্টরে একটি নতুন উপাদান যোগ (Add) করলেই হবে। সুতরাং বলা যায় পলিমরফিজম পদ্ধতির জন্য অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর ক্ষেত্রে প্রোগ্রামিং এর গতি অনেক বেড়েছে, প্রোগ্রামের মেইনটেন্যান্স ও পুনঃব্যবহারযোগ্যতা অনেক সহজ ও উন্নত হয়েছে। ফলে ত্রুটিমুক্ত (robust) সফটওয়্যার ডেভেলপমেন্ট ও অনেক সহজ হয়েছে।

৮.৫ জাভায় অ্যাবস্ট্রাক্ট ক্লাস (Abstract Class in Java) ঃ

যদি কোন ক্লাসে এক বা একাধিক অ্যাবস্ট্রাক্ট মেথড (Abstract method) থাকে অথবা ক্লাসটি যদি কোন অ্যাবস্ট্রাক্ট ক্লাসকে ইনহেরিট করে তবে তাকে অ্যাবস্ট্রাক্ট ক্লাস (Abstract class) বলে। অ্যাবস্ট্রাক্ট ক্লাসের মূল উদ্দেশ্য হল- ক্লাস হায়ারারকী (Hierarchy) এর appropriate সুপার ক্লাস মেইনটেইন করা। অ্যাবস্ট্রাক্ট ক্লাসগুলো real world অবজেক্টকে পূর্ণাঙ্গভাবে প্রকাশের জন্য পর্যাপ্ত নয়।



চিত্র ঃ ক্লাস হায়ারারকী (অ্যাবস্ট্রাক্ট ক্লাস)

উপরের চিত্রটি লক্ষ করলে দেখা যাবে যে, 'Shape' '2D Shape' এবং '3D Shape' ইত্যাদি ক্লাসগুলোর মাধ্যমে কোন ধরনের '2D Shape' কিংবা '3D Shape' draw করতে হবে তা পূর্ণাঙ্গভাবে প্রকাশ করা যাচ্ছে না, তাই 'Shape' '2D Shape' এবং '3D Shape' ক্লাসগুলো অ্যাবস্ট্রাক্ট ক্লাস (Abstract class)। অন্যদিকে 'Square' 'Circle' 'Triangle' কিংবা 'Cube' 'Sphere' ও 'Cylinder' ক্লাসগুলো Concrete class। কারণ এগুলো ব্যবহার করে কোন ধরনের অবজেক্ট Draw করা হবে তা পূর্ণাঙ্গভাবে প্রকাশ পাচ্ছে। যে সকল ক্লাসের মাধ্যমে অবজেক্ট সম্পর্কে পূর্ণাঙ্গ তথ্য পাওয়া যায় তাদেরকে কনক্রিট ক্লাস (Concrete class) বলে।

অ্যাবস্ট্রাক্ট মেথডের কোন বডি থাকে না। অ্যাবস্ট্রাক্ট মেথডকে ডিক্লেয়ার করার নিয়ম হল-

```
abstract ReturnType Method Name ();
```

উদাহরণ ঃ

```
abstract void shape ();
```

অ্যাবস্ট্রাক্ট মেথড বিশিষ্ট কোন সুপার ক্লাস সাব ক্লাসে ইনহেরিট করতে হলে আবশ্যিকভাবে ডিরাইভড ক্লাসে ওভাররাইডিং মেথড ব্যবহার করতে হয়। অর্থাৎ ডিরাইভড ক্লাসে পুনরায় উক্ত অ্যাবস্ট্রাক্ট মেথড বর্ণনা করতে হয়। উল্লেখ্য, abstract কীওয়ার্ড যোগে কোন সুপার ক্লাস এবং তার মেথড অ্যাবস্ট্রাক্ট হিসেবে ঘোষণা করতে হয় এবং অ্যাবস্ট্রাক্ট মেথড সুপার ক্লাসে ঘোষণা করা হলেও সেখানে বর্ণনা করা যায় না। নিম্নে অ্যাবস্ট্রাক্ট সুপার ক্লাসের একটি প্রোগ্রাম দেয়া হল ঃ

```
abstract class Student
{
public int Roll;
public String Name;
abstract void GetData ( );
}
class Result extends Student
{
float Mark;
void GetData( )
{
```

```
Roll=96008;
Name = "Mahfuz";
Mark=80.25f;
}
void Display( )
{
System.out.println("Roll is" +Roll);
System.out.println("Name is" +Name);
System.out.println("Mark is" +Mark);
}
}
class AbstractExample
{
public static void main(String args[])
{
Result R=new Result( );
R.GetData ();
R.Display( );
}
}
```

৮.৬ ফাইনাল কীওয়ার্ড (final Keyword) ঃ

ফাইনাল (final) কীওয়ার্ড অনেকটা কন্সট্যান্ট ভেরিয়েবলের ন্যায় কাজ করে। অর্থাৎ final ভেরিয়েবলে একবার কোন ডাটা সংরক্ষণ করা হলে এটি আর কখনো পরিবর্তন করা যায় না। যেহেতু final মেথডের বডি পরিবর্তন হওয়ার কোন সম্ভাবনা নেই। তাই কম্পাইলার মেথড কল এর স্থলে মেথড বডি বা ডেফিনিশন নিয়ে আসে। এতে প্রোগ্রামের দক্ষতা বহুলাংশে বৃদ্ধি পায়। কোন একটি ক্লাসকে final করা হলে একে আর extend করা যায় না। জাভা ক্লাসে কোন ভেরিয়েবলকে ফাইনাল হিসেবে ডিক্লারেশনের জন্য ডাটা টাইপের পূর্বে 'final' কীওয়ার্ড ব্যবহার করা হয়। ফাইনাল মেথডকে ডিক্লেয়ার করার নিয়ম হল–

```
final ReturnType VariableName = Value;
```

উদাহরণ ঃ

```
final void gravity = 9.8;
```

ফাইনাল মডিফায়ারের ক্ষেত্রে উল্লেখ্য যে,

–ফাইনাল (final) অবজেক্টের রেফারেন্স ভেরিয়েবল পরিবর্তন করা যাবে না।

– ফাইনাল (final) অবজেক্টের ভিতর অবস্থিত ডাটার পরিবর্তন করা যাবে ।

– ফাইনাল (final) মেথডকে কখনো ওভাররাইড (Override) করা যাবে না।

৮.৭ ইনহেরিটেন্স ও পলিমরফিজম ব্যবহার করে প্রোগ্রাম (Programs Using Inheritance & Polymorphism)

[উক্ত অধ্যায়ের পূর্বের টপিকসগুলোতে প্রোগ্রাম লেখা হয়েছে।]

অনুশীলনী-৮

অতি সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। সুপার ক্লাস ও সাব ক্লাস কাকে বলে?

উত্তরঃ **সুপার ক্লাস (Super class) ঃ** ইনহেরিটেন্স প্রক্রিয়ায় পূর্বে ডিফাইনকৃত যে এক বা একাধিক ক্লাস হতে বৈশিষ্ট্য ইনহেরিট করা হয় সে বা সে সকল ক্লাসকে সুপার ক্লাস (Super class) বলে।

সাব ক্লাস (Sub class) ঃ ইনহেরিটেন্স প্রক্রিয়ায় পূর্বে ডিফাইনকৃত এক বা একাধিক ক্লাস হতে বৈশিষ্ট্য ইনহেরিট করে নতুন যে ক্লাস তৈরি হয় সেই ক্লাসকে সাব ক্লাস (sub class) বলে।

২। জাভায় ইনহেরিটেন্স কত প্রকার ও কী কী?

উত্তরঃ সুপার ক্লাস ও সাব ক্লাসের সংখ্যা এবং সাব ক্লাস গঠনের কৌশলের ভিত্তিতে জাভা ভাষায় ইনহেরিটেন্স প্রক্রিয়াকে মূলত তিন ভাগে ভাগ করা হয় ঃ

- সিঙ্গেল ইনহেরিটেন্স (Single Inheritance)।
- মাল্টিলেভেল ইনহেরিটেন্স (Multilevel Inheritance)।
- হায়ারারকিক্যাল ইনহেরিটেন্স (Hierarchical Inheritance)।

৩। সিঙ্গেল ইনহেরিটেন্স কাকে বলে?

উত্তরঃ যখন কোন সাব ক্লাস কেবল একটিমাত্র বেস ক্লাস থেকে এক বা একাধিক বৈশিষ্ট্য ইনহেরিট করে, তখন সেই প্রক্রিয়াকে সিঙ্গেল ইনহেরিটেন্স (Single Inheritance) বলা হয়।

৪। মাল্টিলেভেল ইনহেরিটেন্স কাকে বলে?

উত্তরঃ যখন কোন সাব ক্লাস অপর কোন সুপার ক্লাসের সাব ক্লাস থেকে এক বা একাধিক বৈশিষ্ট্য ইনহেরিট করে, তখন সেই প্রক্রিয়াকে মাল্টিলেভেল ইনহেরিটেন্স (Multilevel Inheritance) বলা হয়।

৫। হায়ারারকিক্যাল ইনহেরিটেন্স কী?

উত্তরঃ যখন একটি মাত্র সুপার ক্লাস হতে দুই বা ততোধিক সাব ক্লাস গঠন করা হয়, তখন সেই প্রক্রিয়াকে হায়ারারকিক্যাল ইনহেরিটেন্স (Hierarchical Inheritance) বলা হয়।

৬। ওভাররিডেন ও ওভাররাইডিং মেথড বলতে কী বুঝায়?

উত্তরঃ কোন মেথড যদি সুপার ক্লাস এবং সাব-ক্লাস দুটিতেই বিদ্যমান থাকে তবে সুপার ক্লাস-এর মেথডটিকে ওভাররিডেন মেথড (Overridden method) বলে। আর এর করেসপন্ডিং (Correspondin) সাব-ক্লাস-এর মেথডটিকে ওভাররাইডিং মেথড (Overriding method) বলে।

৭। অ্যাবস্ট্রাক্ট ক্লাস কী?

উত্তরঃ যদি কোন ক্লাসে এক বা একাধিক অ্যাবস্ট্রাক্ট মেথড (Abstract method) থাকে অথবা ক্লাসটি যদি কোন অ্যাবস্ট্রাক্ট ক্লাসকে ইনহেরিট করে তবে তাকে অ্যাবস্ট্রাক্ট ক্লাস (Abstract class) বলে।

৮। ফাইনাল (final) কীওয়ার্ড কী?

উত্তরঃ ফাইনাল (final) কীওয়ার্ড অনেকটা কন্সট্যান্ট ভেরিয়েবলের ন্যায় কাজ করে। অর্থাৎ final ভেরিয়েবলে একবার কোন ডাটা সংরক্ষণ করা হলে এটি আর কখনো পরিবর্তন করা যায় না।

৯। ইনহেরিটেন্স ব্যবহারের সুবিধা কী কী?

উত্তরঃ ইনহেরিটেন্স প্রক্রিয়ার মাধ্যমে ক্লাসের পুনঃব্যবহার (reusability) প্রক্রিয়া বাস্তবায়ন করা হয়। এতে করে প্রোগ্রামের সময় খরচ ও ঝামেলা কমার পাশাপাশি প্রোগ্রামের সৌন্দর্য ও গ্রহণযোগ্যতা বৃদ্ধি পায়।

সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। মাল্টিলেভেল ইনহেরিটেন্স এর ব্লক চিত্র অঙ্কন করে ফরম্যাট লেখ।

উত্তর সংকেত ঃ ৮.২ নং অনুচ্ছেদ দ্রষ্টব্য।

২। হায়ারারকিক্যাল ইনহেরিটেন্স এর ব্লকচিত্রসহ ফরম্যাট লেখ।

উত্তর সংকেত ঃ ৮.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। সিঙ্গেল ইনহেরিটেন্স ও মাল্টিলেভেল ইনহেরিটেন্স এর মধ্যে পার্থক্য লেখ।

উত্তর সংকেত ঃ ৮.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। ওভাররিডেন মেথডের বৈশিষ্ট্যসমূহ উল্লেখ কর।

উত্তর সংকেত ঃ ৮.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। উদাহরণসহ পলিমরফিজম প্রক্রিয়াটি ব্যাখ্যা কর।

উত্তর সংকেত ঃ ৮.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। উদাহরণসহ অ্যাবস্ট্রাক্ট ক্লাস ও কনক্রিট ক্লাস ব্যাখ্যা কর।

উত্তর সংকেত ঃ ৮.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৭। ফাইনাল মডিফায়ারের ক্ষেত্রে বিবেচ্য বিষয়সমূহ কী কী?

উত্তর সংকেত ঃ ৮.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

রচনামূলক প্রশ্নোত্তর ঃ

১। সুপার ক্লাস ও সাব ক্লাস ব্যবহার করে একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ৮.২ নং অনুচ্ছেদ বা ব্যবহারিক অংশ দ্রষ্টব্য।

২। মাল্টিলেভেল ইনহেরিটেন্স ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ৮.২ নং অনুচ্ছেদ বা ব্যবহারিক অংশ দ্রষ্টব্য।

৩। হায়ারারকিক্যাল ইনহেরিটেন্স প্রক্রিয়াটি একটি প্রোগ্রামের মাধ্যমে ব্যাখ্যা কর।

উত্তর সংকেত ঃ ৮.২ নং অনুচ্ছেদ বা ব্যবহারিক অংশ দ্রষ্টব্য।

৪। ওভাররিডেন মেথড ব্যবহার করে একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ৮.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। অ্যাবস্ট্রাক্ট ক্লাস ব্যবহার করে একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ৮.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

অধ্যায় ০৯

প্যাকেজ ও ইন্টারফেস
Packages and Interfaces

৯.০ ভূমিকা (Introduction) ঃ

জাভাতে একটি এপ্লিকেশন হচ্ছে কতগুলো পরস্পর সম্পর্কিত ক্লাসের সমষ্টিমাত্র। এ ল্যাংগুয়েজে প্রতিটি Executable code কে কোন না কোন ক্লাসের অন্তর্ভুক্ত হতে হয়। ক্লাসের ভিতর ভেরিয়েবল, প্রপার্টি, অ্যাট্রিবিউট ও মেথড থাকে। ক্লাসগুলো এদের বৈশিষ্ট্য অনুসারে বিভিন্ন প্যাকেজ (Package) এর ভিতর রাখা হয়। প্যাকেজগুলো হচ্ছে অনেকটা অপারেটিং সিস্টেমের ফোল্ডার (Folder) এর মত। জাভা প্রোগ্রামকে Set of package হিসেবে Organize করা হয়েছে। জাভার ফাউন্ডেশন ক্লাসগুলোকে বিভিন্ন প্যাকেজে সাজিয়ে উপস্থান করা হয়েছে যাতে ইউজার খুব সহজে নির্দিষ্ট প্যাকেজে গিয়ে নির্দিষ্ট ক্লাসটি খুঁজে নিতে পারে। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর আরেকটি চমৎকার আকর্ষণীয় ও প্রয়োজনীয় বিষয় হচ্ছে ইন্টারফেস (Interface) । ইন্টারফেস জাভা প্যাকেজের ভিতরে অবস্থান করে। উক্ত অধ্যায়ে আমরা প্যাকেজ ও ইন্টারফেস এর ব্যবহার এবং তাদের অন্যান্য গুরুত্বপূর্ণ বিষয়গুলো নিয়ে আলোচনা করব।

৯.১ সিনটেক্সসহ প্যাকেজের সংজ্ঞা (Define Package with Syntax) ঃ

প্যাকেজ (Package) ঃ প্যাকেজ (Package) হল পরস্পর সংশ্লিষ্ট কতগুলো ক্লাস ও ইন্টারফেসের সমষ্টি। প্যাকেজের মাধ্যমে প্রোগ্রামে ব্যবহৃত ক্লাস ও ইন্টারফেস গুলোকে Organize অবস্থায় ছোট ছোট ফোল্ডারের মাধ্যমে সাজিয়ে রাখা যায়। ফলে পরবর্তীতে খুব সহজেই নির্দিষ্ট প্যাকেজে গিয়ে নির্দিষ্ট ক্লাসটি খুঁজে পাওয়া যায়।

সিনটেক্স (Syntax) ঃ প্যাকেজ (Package) ডিক্লারেশনের ফরম্যাট হল–

```
package PackageName;
```

এখানে 'package' হল কী ওয়ার্ড এবং 'PackageName' হল ইউজার কর্তৃক প্রদত্ত প্যাকেজের নাম। প্যাকেজের নাম একটি আইডেন্টিফায়ার, তাই আইডেন্টিফায়ার নামকরণের নিয়মানুযায়ী যে কোন বৈধ নামই প্যাকেজের নাম হতে পারে। প্যাকেজের নাম ছোট হাতের অক্ষরে হবে।

উদাহরণ ঃ

```
package mypackage;
package bird; ইত্যাদি।
```

প্যাকেজের প্রকারভেদ (Classification of package) ঃ জাভায় দুই ধরনের প্যাকেজ (Package) ব্যবহার করা হয়। যথা ঃ

- জাভা সিস্টেম প্যাকেজ (Java System Package) ও
- ইউজার ডিফাইন্ড প্যাকেজ (User Defined Package)

জাভা সিস্টেম প্যাকেজ (Java system package) ঃ

জাভায় বিল্ট-ইন ক্লাসগুলো যে সকল প্যাকেজে সুশৃঙ্খলভাবে সাজানো থাকে তাদেরকে জাভা সিস্টেম প্যাকেজ (Java system package) বলে । জাভা সিস্টেম প্যাকেজ (Java system package) হল অনেকটা সি/সি++ এর হেডার ফাইলের মত। জাভায় বহুল ব্যবহৃত সিস্টেম প্যাকেজ বা বিল্ট-ইন প্যাকেজগুলো হচ্ছে–

জাভা সিস্টেম প্যাকেজ		
java. lang	java. awt	java. math
java. io	java. net	java. rmi
java. utitl	java. swing	java. security
java. applet	java. beans	java. text ইত্যাদি।

কোন প্রোগ্রামে জাভার সিস্টেম প্যাকেজ বা প্যাকেজের কোন মেথডকে ব্যবহার করার জন্য 'import' স্টেটমেন্ট ব্যবহার করতে হয়। সাধারণত প্রোগ্রামের শুরুতেই কোন সিস্টেম প্যাকেজকে কল (Call) করতে হয়। প্রোগ্রামের শুরুতে import স্টেটমেন্ট ব্যবহার করে কোন প্যাকেজকে কল করলে প্রোগ্রামের যে কোন স্থানে সরাসরি উক্ত প্যাকেজের অন্তর্গত যে কোন ক্লাস ও তার মেথড ব্যবহার করা যায়। জাভা সিস্টেম প্যাকেজ কল করার ফরম্যাট হল–

```
import java.*;
```

আবার জাভার নির্দিষ্ট প্যাকেজ 'lang' এর 'Math' ক্লাসের যে কোন মেথড একসেস করার স্টেটমেন্ট হচ্ছে–

```
import java.lang.Math.*;
```

ইউজার ডিফাইন্ড প্যাকজ (User defined package) ঃ ইউজার পরবর্তীতে পুনঃব্যবহারের উদ্দেশ্যে যখন এক বা একাধিক ক্লাসকে কোন প্যাকেজের অধীনে সংরক্ষণ করেন তখন সেই প্যাকেজকে ইউজার ডিফাইন্ড প্যাকেজ (User Defined Package) বলে।

সিনট্যাক্স (Syntax) ঃ ইউজার ডিফাইন্ড প্যাকেজ তৈরির ফরম্যাট হল–

```
package PackageName;
```

উদাহরণ ঃ

```
package mypackage;
```

উল্লেখ্য যে,

নিয়মানুযায়ী প্যাকেজের নাম (PackageName) লেখার ক্ষেত্রে ছোট হাতের অক্ষর ব্যবহার করতে হয়। এতে করে ক্লাসের নাম ও প্যাকেজের নাম এর মধ্যে কোন Conflict তৈরি হবে না।

একটি প্যাকেজের অধীনে একই নামে দুটি ক্লাস থাকতে পারবেনা। তবে ভিন্ন প্যাকেজে একই নামের দুটি ক্লাস থাকতে পারে।

```
package Pack;
public class PackExample
{
int Roll;
String Name;
float Mark;
```

```
public void GetData()
{
Roll=96002;
Name="Munny";
Mark=85.75f;
}
public void Display()
{
System.out.println("Roll is : "+Roll);
System.out.println("Name is : "+Name);
System.out.println("Mark is : "+Mark);
}
}
```

উপরোক্ত প্যাকেজ প্রোগ্রাম ফাইল কম্পাইল করলে কম্পাইলার একই ডিরেক্টরিতে PackExample নামে একটি ক্লাস ফাইল তৈরি করবে।

এবার উপরোক্ত প্যাকেজ ব্যবহারকারী প্রোগ্রামটি নিম্নে দেয়া হল। প্রোগ্রামটি ব্যবহারের জন্য উপরোক্ত প্যাকেজটি যে লোকেশনে সেভ করা হয়েছে এই প্রোগ্রামটিও সেই একই লোকেশনে সেভ করতে হবে।

```
import Packs.PackExample;
class PackUse
{
public static void main(String args[])
{
PackExample P=new PackExample();
P.GetData();
P.Display();
}
}
```

৯.২ প্যাকেজ ডিক্লারেশনের প্রয়োজনীয়তা (Necessity of Package Declaration) ঃ

- প্যাকেজের মাধ্যমে classগুলোকে organize অবস্থায় ছোট ছোট folder এর ভিতর সাজিয়ে রাখা যায় ও নির্দিষ্ট class fileটি খুঁজে পেতে সমস্যা হয় না।
- নামজনিত সমস্যাগুলো হতেও মুক্ত থাকা যায়। strategical কারণে দুটি same নামের class একই folder এ রাখা যায় না। package অনুসারে সাজিয়ে আমরা এই সমস্যা থেকে মুক্তি পেতে পারি।
- Package এর ভিতর class রেখে আমরা default access modifier এর সুবিধাসমূহ পেতে পারি।
- Package name এর মাধ্যমে class identify করা সম্ভব।

৯.৩ বিভিন্ন ধরনের প্যাকেজের ফাংশন (Describe the Function of Different Packages) ঃ

জাভা ভাষায় খুবই শক্তিশালী ক্লাস ও ইন্টারফেস সমৃদ্ধ বেশ কিছু সিস্টেম প্যাকেজ রয়েছে। নিম্নে এসকল প্যাকেজসমূহ আলোচনা করা হল–

অ্যাপলেট (Java. applet) ঃ সাধারণত ইন্টারনেট অ্যাপ্লিকেশনের ক্ষেত্রে অ্যাপলেট প্যাকেজটি ব্যবহৃত হয়। অ্যাপলেট (Applet) প্যাকেজে ব্যবহৃত মেথডগুলো হচ্ছে–

init()
start()
stop () এবং
destroy () ইত্যাদি।

এডব্লিউটি (Java. awt) ঃ AWT মানে হল Abstract Windowing Toolkit. AWT প্যাকেজের ভেতর GUI (Graphical User Interface) সম্পর্কিত প্রয়োজনীয় ক্লাস ও ইন্টারফেসসমূহ বিদ্যমান। এতে প্রায় ৬৪টি ক্লাস ও ১৪টি ইন্টারফেস রয়েছে। তাদের মধ্যে উল্লেখযোগ্য কয়েকটি হল–

Java. awt. color
Java. awt. font
Java. awt. event
Java. awt. image ইত্যাদি।

বিনস (Java. beans) ঃ জাভা বিন হচ্ছে একটি সফটওয়্যার কম্পোনেন্ট যা বিভিন্ন পরিবেশে পুনঃব্যবহারের জন্য ডিজাইন করা হয়েছে। যে কোন সহজ ফাংশন অথবা জটিল ফাংশন উভয় ধরনের কাজই জাভা বিন এর মাধ্যমে করা সম্ভব। জাভা বিনস এ ১৭টি ক্লাস ও ৮টি ইন্টারফেস রয়েছে। তাদের মধ্যে উল্লেখযোগ্য কয়েকটি হল–

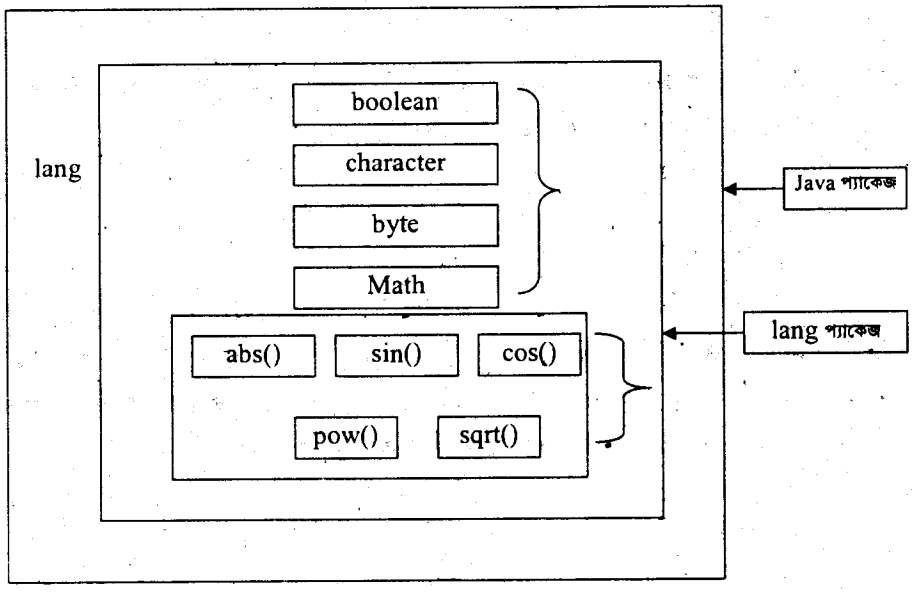
Bean Descriptor
Encoder
Event handler
Expression
Introspector ইত্যাদি।

স্যুইং (Java. swing) ঃ AWT এর সীমাবদ্ধতাকে দূর করার জন্যই জাভা স্যুইং এর উৎপত্তি। স্যুইং সাধারণত দুই ধরনের কী ফিচার নিয়ে কাজ করে। যেমন– Lightweight component ও Pluggable look & feel. জাভা স্যুইং এ ১৪৭টি ক্লাস ও ২২টি ইন্টারফেস আছে। উল্লেখযোগ্য কিছু swing প্যাকেজ হল–

Javax. swing. border
Javax. swing. event
Javax. swing. preview
Javax. swing. text
Javax. swing. tree ইত্যাদি।

৯.৪ প্যাকেজের স্ট্রাকচার (Structure of Package) ঃ

প্যাকেজের মধ্যে বিভিন্ন ক্লাস ও ইন্টারফেসসমূহ বিভিন্ন সাব ক্লাস, অবজেক্ট, মেথড ইত্যাদি সমন্বয়ে হায়ারারকিক্যাল স্ট্রাকচারে সজ্জিত থাকে। একটি প্যাকেজের অধীনে আবার এক বা একাধিক সাব প্যাকেজ থাকতে পারে। নিম্নে জাভা সিস্টেম প্যাকেজ java.lang এর হায়ারারকিক্যাল স্ট্রাকচার দেখান হল।



৯.৫ সিনটেক্সসহ ইন্টারফেস (Interface with Syntax) ঃ

ইন্টারফেস হল এক বিশেষ ধরনের ক্লাস যাতে কতগুলো ভেরিয়েবল এবং মেথডের ঘোষণা থাকে। ইন্টারফেসে মেম্বার ভেরিয়েবলগুলো স্ট্যাটিক ও কনস্ট্যান্ট প্রকৃতির। সুতরাং ইন্টারফেস বডিতে সেগুলোর ঘোষণার সময়ই মান নির্ধারণ করে দিতে হয় এবং ইন্টারফেস বডিতে মেম্বার ফাংশনগুলোর কেবল ঘোষণা থাকে, সাব ক্লাসে সেগুলোর বিস্তারিত বর্ণনা থাকে।

সিনটেক্স (Syntax) ঃ ইন্টারফেস ডিক্লারেশনের সাধারণ ফরম্যাট হল ঃ

```
interface InterfaceName
{
// Member Variable
// Member Function
}
```

উদাহরণ ঃ

```
interface Student
{
static final int Roll=8404;          //static final member
final static String Name= "Mitali";      //static final member
void Display( );                     //interface method decleration
}
```

৯.৬ ইন্টারফেস বাস্তবায়ন প্রক্রিয়া (Interface Implementation) ঃ

ইন্টারফেস (Interface) এর মাধ্যমে অনেক বাস্তব সম্মত সমস্যার (Real world problem) সমাধান করা যায়। ক্লাসকে 'extends' আর ইন্টারফেসকে 'implements' করা হয়। একটি ক্লাস শুধুমাত্র একটি ক্লাসকেই extends করতে পারে কিন্তু একটি ক্লাস অগণিত ইন্টারফেসকে implements করতে পারে।

মূলত সাব ক্লাস বা ক্লাসসমূহে কতগুলো নির্ধারিত ভেরিয়েবল ও মেথড স্থানান্তরের জন্য ইন্টারফেস ব্যবহৃত হয়। ইন্টারফেস বাস্তবায়নে ফরম্যাট হল–

```
class SubclassName implements InterfaceName
{
// Sub class's Variable Declaration
// Sub class's Method Definition
// Interface's Method Definition
}
```

এখানে class ও implements দুইটি জাভা কীওয়ার্ড, SubClassName কাঙ্ক্ষিত সাব ক্লাসের নাম এবং InterfaceName পূর্বে ঘোষিত কোন ইন্টারফেসের নাম। ইন্টারফেস বাস্তবায়নকারী সাব ক্লাসে ইন্টারফেস বডিতে ঘোষিত মেম্বার ফাংশনগুলো বর্ণনাসহ নিজস্ব মেম্বার ভেরিয়েবলগুলোর ঘোষণা এবং মেম্বার ফাংশনগুলো বর্ণনা থাকে। সাধারণত ইন্টারফেসের জন্য কোন অবজেক্ট ঘোষণা করা যায় না। সাব ক্লাস অবজেক্টের মাধ্যমেই ইন্টারফেস মেম্বার এক্সেস করা হয়। নিম্নে ইন্টারফেস প্রক্রিয়া বাস্তবায়নের একটি নমুনা প্রোগ্রাম দেয় হল ঃ

```
interface Student
{
static final int Roll=8404;          //static final member
 static final String Name="Mitali";      //static final member
void Display( );                     //Interface method decleared
}
class Result implements Student
{
public float Mark;
public void Display()                //Interface method defined
{
System.out.println("Roll is : "+Roll);
System.out.println("Name is :"+Name);
}
public void GetMark()                //own method defined
{
Mark=65.45f;
}
public void ShowMark()               //own method defined
{
System.out.println("Mark is :"+Mark);
```

```
}
}
public class interf
{
public static void main(String args[])
{
Result R=new Result();
R.Display();
R.GetMark();
R.ShowMark();
}
}
```

৯.৭ মাল্টিপল ইন্টারফেস (Multiple Interface) ঃ

যদি একই সাব ক্লাসে দুই বা ততোধিক ইন্টারফেস বাস্তবায়ন করা হয় তবে তাকে মাল্টিপল ইন্টারফেস (Multiple interface) বলে। মাল্টিপল ইন্টারফেস বাস্তবায়নের ক্ষেত্রে সাব ক্লাস হেডার লাইনে 'implements' কী ওয়ার্ডের পর ইন্টারফেসগুলোর নাম এবং ইন্টারফেসগুলোর মাঝে কমা দিতে হয়।

সিনটেক্স (Syntax) ঃ

```
interface student
{
// interface member and methods
}
interface exam
{
// interface variable or methods
}
class result implements student, exam // মাল্টিপল ইন্টারফেস
{
}
```

নিম্নে মাল্টিপল ইন্টারফেস ব্যবহার করে একটি প্রোগ্রাম লেখা হল,

```
interface Student
{
String Name="Tonny";          //static final member
}
interface Exam
{
int Roll=96004;               //static final member
void Display( );              //Interface method decleared
}
```

```
class Result implements Student, Exam
{
public float Mark;
public void Display()              //Interface method defined
{
System.out.println("Roll is : "+Roll);
System.out.println("Name is :"+Name);
}
public void GetMark()              //own method defined
{
Mark = 87.5f;
}
public void ShowMark()             //own method defined
{
System.out.println("Mark is :"+Mark);
}
}
public class mulinterf
{
public static void main(String args[])
{
Result R=new Result();
R.Display();                       //Calling interface Method
R.GetMark();                       //Calling Derived Class Method
R.ShowMark();                      //Calling Derived Class Method
}
}
```

৯.৮ প্যাকেজ ও ইন্টারফেস ব্যবহার করে প্রোগ্রাম (Programs using package and interface) ঃ

[উক্ত অধ্যায়ের পূর্বের টপিকসগুলোতে এ সংক্রান্ত প্রোগ্রাম লেখা হয়েছে।]

অনুশীলনী-৯

অতি সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। প্যাকেজ বলতে কী বোঝায়?

উত্তরঃ প্যাকেজ হল কতকগুলো ক্লাস ও ইন্টারফেসের সমষ্টি। জাভা ভাষায় সি বা সি++ এর মত হেডার ফাইল ব্যবহারের পরিবর্তে প্রয়োজনীয় কতকগুলো ক্লাস ও ইন্টারফেস একত্রে ঘোষণা করা হয়, যা অন্য যে কোন প্রোগ্রাম বা সিস্টেমে ব্যবহার করা যায়। জাভা প্রোগ্রামে দুই ধরনের প্যাকেজ ব্যবহার করা যায়, যথা-(ক) জাভা সিস্টেম প্যাকেজ। (খ) ইউজার ডিফাইন্ড প্যাকেজ।

২। Built-in package গুলো কী কী?

উত্তরঃ জাভা প্রোগ্রামে ব্যবহৃত Built-in package গুলো হল- java. lang, java. io, java. util, java. applet, java. awt এবং java. net.

৩। User-defined package ঘোষণার পদ্ধতি কী?

উত্তরঃ জাভা ভাষায় User-defined package তৈরি করার জন্য package কীওয়ার্ডের পরে ঐ প্যাকেজের নাম দিতে হয়, অতঃপর সেমিকোলন দিতে হয়। প্যাকেজ ঘোষণা সাধারণত প্রোগ্রামের উপরের অংশে অর্থাৎ অন্য কোন ক্লাস ঘোষণার পূর্বে থাকে।

৪। ইনার ক্লাস ও আউটার ক্লাস কী?

উত্তরঃ একটি ক্লাসের ভিতর অন্য একটি ক্লাসকে ডিফাইন করাকে ইনার ক্লাস এবং যে ক্লাসের ভিতর ইনার ক্লাসকে ডিফাইন করা হয় তাকে আউটার ক্লাস বলে।

৫। Nested package কী?

উত্তরঃ একটি জাভা প্যাকেজের অধীনে এক বা একাধিক প্যাকেজ বর্ণনা করা যায়। এরূপ একটি Package এর মধ্যে বিদ্যমান এক বা একাধিক সাব প্যাকেজের সমষ্টিকে Nested package বলে।

৬। জাভা বিনস কী?

উত্তরঃ জাভা বিন হচ্ছে একটি সফটওয়্যার কম্পোনেন্ট যা বিভিন্ন পরিবেশে পুনঃব্যবহারের জন্য ডিজাইন করা হয়েছে। যে কোন সহজ ফাংশন অথবা জটিল ফাংশন উভয় ধরনের কাজই জাভা বিন এর মাধ্যমে করা সম্ভব। জাভা বিনস এ ১৭টি ক্লাস ও ৮টি ইন্টারফেস রয়েছে।

৭। উল্লেযোগ্য কয়েকটি স্যুইং প্যাকেজের নাম লেখ।

উত্তরঃ উল্লেখযোগ্য কিছু swing প্যাকেজ হল-

Javax. swing. border
Javax. swing. event
Javax. swing. preview
Javax. swing. text
Javax. swing. tree ইত্যাদি।

৮। ইন্টারফেস বলতে কী বুঝায়? এর বৈশিষ্ট্য কী?

উত্তরঃ ইন্টারফেস হল এক বিশেষ ধরনের ক্লাস যাতে কতগুলো ভেরিয়েবল এবং মেথডের ঘোষণা থাকে। ইন্টারফেসে মেম্বার ভেরিয়েবলগুলো স্ট্যাটিক ও কনস্ট্যান্ট প্রকৃতির। সুতরাং ইন্টারফেস বডিতে সেগুলোর ঘোষণার সময়ই মান নির্ধারণ করে দিতে হয় এবং ইন্টারফেস বডিতে মেম্বার ফাংশনগুলোর কেবল ঘোষণা থাকে, সাব ক্লাসে সেগুলোর বিস্তারিত বর্ণনা থাকে।

৯। মাল্টিপল ইন্টারফেস কাকে বলে?

উত্তরঃ যদি একই সাব ক্লাসে দুই বা ততোধিক ইন্টারফেস বাস্তবায়ন করা হয় তবে তাকে মাল্টিপল ইন্টারফেস (Multiple interface) বলে। মাল্টিপল ইন্টারফেস বাস্তবায়নের ক্ষেত্রে সাব ক্লাস হেডার লাইনে 'implements' কী ওয়ার্ডের পর ইন্টারফেসগুলোর নাম এবং ইন্টারফেসগুলোর মাঝে কমা দিতে হয়।

১০। ইউজার ডিফাইন্ড প্যাকেজ কাকে বলে?

উত্তরঃ ইউজার পরবর্তীতে পুনঃব্যবহারের উদ্দেশ্যে যখন এক বা একাধিক ক্লাসকে কোন প্যাকেজের অধীনে সংরক্ষণ করেন তখন সেই প্যাকেজকে ইউজার ডিফাইন্ড প্যাকেজ (User Defined Package) বলে।

১১। জাভা সিস্টেম প্যাকেজ বলতে কী বুঝায়?

উত্তরঃ জাভায় বিল্ট-ইন ক্লাসগুলো যে সকল প্যাকেজে সুশৃঙ্খলভাবে সাজানো থাকে তাদেরকে জাভা সিস্টেম প্যাকেজ (Java system package) বলে । জাভা সিস্টেম প্যাকেজ (Java system package) হল অনেকটা সি/সি++ এর হেডার ফাইলের মত।

১২। ইন্টারফেস বাস্তবায়নের ফরম্যাট লেখ।

উত্তরঃ ইন্টারফেস বাস্তবায়নে ফরম্যাট হল–

```
class SubclassName implements InterfaceName
{
// Sub class's Variable Declaration
// Sub class's Method Definition
// Interface's Method Definition
}
```

➥ *সংক্ষিপ্ত প্রশ্নোত্তর ঃ*

১। উদাহরণসহ প্যাকেজ ও ইন্টারফেস এর বর্ণনা দাও।

উত্তর সংকেত ঃ ৯.১ ও ৯.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

২। প্যাকেজ ডিক্লারেশনের প্রয়োজনীয়তা কী?

উত্তর সংকেত ঃ ৯.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। জাভায় ব্যবহৃত বিভিন্ন ধরনের সিস্টেম প্যাকেজ সংক্ষেপে বর্ণনা কর।

উত্তর সংকেত ঃ ৯.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। জাভা সিস্টেম প্যাকেজের হায়ারারকিক্যাল স্ট্রাকচার দেখাও।

উত্তর সংকেত ঃ ৯.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। স্যাইং কী? স্যাইং এর ফিচারগুলোর নাম লেখ।

উত্তর সংকেত ঃ ৯.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। উদাহরণসহ ইন্টারফেস এর ফরম্যাট উল্লেখ কর।

উত্তর সংকেত ঃ ৯.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৭। মাল্টিপল ইন্টারফেস এর syntax লেখ।

উত্তর সংকেত ঃ ৯.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

➥ *রচনামূলক প্রশ্নোত্তর ঃ*

১। একটি প্রোগ্রাম লিখে ইউজার ডিফাইন্ড প্যাকেজ তৈরি কর।

উত্তর সংকেত ঃ ৯.১ নং অনুচ্ছেদ দ্রষ্টব্য।

২। ইন্টারফেস বাস্তবায়ন প্রক্রিয়াটি একটি প্রোগ্রাম লিখে ব্যাখ্যা কর।

উত্তর সংকেত ঃ ৯.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। মাল্টিপল ইন্টারফেস প্রক্রিয়াটি প্রোগ্রামসহ বর্ণনা কর।

উত্তর সংকেত ঃ ৯.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

অধ্যায় ১০

মাল্টিথ্রেড প্রোগ্রামিং
Multithread programming

১০.০ ভূমিকা (Introduction) ঃ

বর্তমানে প্রচলিত উইন্ডোজ ভিত্তিক অপারেটিং সিস্টেমগুলো (যেমন– উইন্ডোজ ৯৫, উইন্ডোজ ৯৮, উইন্ডোজ XP, উইন্ডোজ 7, উইন্ডোজএনটি, ইত্যাদি) মাল্টিটাস্কিং (Multitasking) অপারেটিং সিস্টেম হিসেবে পরিচিত। মাল্টিটাস্কিং অপারেটিং সিস্টেমগুলোতে একই সাথে একাধিক টাস্ক বা প্রোগ্রাম চালানো যায়। যেমন উইন্ডোজ অপারেটিং সিস্টেমে মিডিয়া প্লেয়ার-এ গান শোনার পাশাপাশি মাইক্রোসফট ওয়ার্ডে লেখা যায়, প্রিন্ট করা যায় ইত্যাদি। সাধারণ সিস্টেমগুলোতে যেখানে একসাথে একটির বেশি প্রসেস নির্বাহ করা যায় না, সেখানে মাল্টিটাস্কিং সিস্টেমের ব্যবহার নিশ্চয়ই সুবিধাজনক। জাভা ভাষা মাল্টিটাস্কিং সিস্টেমের অনুরূপ অপর একটি প্রোগ্রাম পদ্ধতির ব্যবহার অনুমোদন করে যা মাল্টিথ্রেডিং (Multithreading) নামে পরিচিত। মাল্টিথ্রেডিং পদ্ধতিতে একই সাথে একই সিস্টেমের একাধিক প্রসেস (সাব-প্রোগ্রাম) চালনা করা যায়। যেমন– মাইক্রোসফট ওয়ার্ডে একটি ডকুমেন্টে প্রিন্ট করার পাশাপাশি অপর একটি ডকুমেন্টে লেখা যায়। মাল্টিটাস্কিং পদ্ধতি অপেক্ষা মাল্টিথ্রেডিং পদ্ধতির পার্থক্য হল মাল্টিটাস্কিং পদ্ধতিতে যেখানে এক সাথে ভিন্ন ভিন্ন সিস্টেমের প্রোগ্রাম চালনা করা যায়, মাল্টিথ্রেডিং পদ্ধতিতে সেখানে একই সাথে কেবল একই সিস্টেমের একাধিক প্রসেস (সাব- প্রোগ্রাম) পরিচালনা করা যায়।

১০.১ সিনটেক্সসহ মাল্টিথ্রেড প্রোগ্রামিং (Multithread Programming with Syntax) ঃ

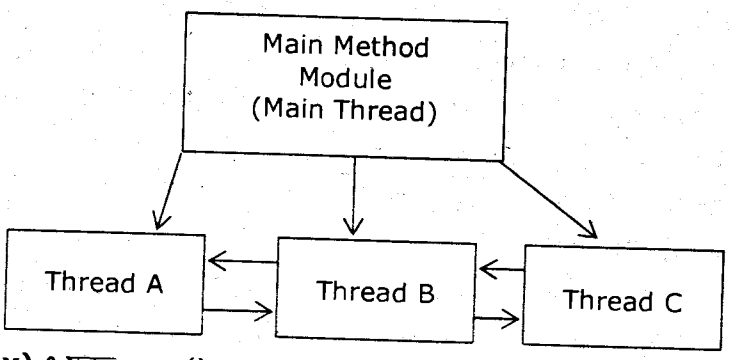
থ্রেড (Thread) ঃ একই সময়ে এক সাথে একাধিক কাজ করার প্রক্রিয়াকে থ্রেড (Thread) বলে। যেমন– মিডিয়া প্লেয়ারে গান শোনা, এমএসওয়ার্ডে ডকুমেন্ট কম্পোজ করা, ও প্রিন্টারে প্রিন্ট করা, এই তিনটি কাজ একই সময়ে এক সাথে করার প্রক্রিয়াই হচ্ছে থ্রেড। যে কোন জাভা প্রোগ্রামে এক বা একাধিক থ্রেড থাকে। যদি কোন প্রোগ্রামে একটি মাত্র থ্রেড থাকে তবে তাকে সিঙ্গেল থ্রেডেড প্রোগ্রাম (Single threaded program) বলে। main () মেথড বিশিষ্ট প্রোগ্রামসমূহ সিঙ্গেল থ্রেডেড প্রোগ্রামের অন্তর্ভুক্ত। এরূপ প্রোগ্রামের স্টেটমেন্টসমূহ পর্যায়ক্রমে একটির পর একটি সম্পাদিত হয়। সিঙ্গেল থ্রেডেড প্রোগ্রামে কেবল একটি মাত্র কন্ট্রোল ফ্লো (Control flow) থাকে।

মূলত তিনটি স্থানে জাভার থ্রেডের বিষয়গুলো অবস্থান করে। যথা ঃ

- Java. lang. Thread ক্লাসে
- Java. lang. object ক্লাসে এবং
- জাভা ভার্চুয়াল মেশিন (JVM) এ।

মাল্টিথ্রেড প্রোগ্রামিং (Multithread programming) ঃ যখন কোন প্রোগ্রামে একই সময়ে এক সাথে একাধিক থ্রেড সক্রিয় (active) থাকে তখন তাকে মাল্টিথ্রেড প্রোগ্রামিং (Multithread programming) বলে।

মাল্টিথ্রেডিং প্রক্রিয়ায় যেহেতু একই সময়ে একাধিক থ্রেড সক্রিয় থাকে তাই এরূপ প্রোগ্রামে একাধিক কন্ট্রোল ফ্লো বিদ্যমান থাকে। ফলে প্রোগ্রামে কাছাকাছি সময়ে কখনও এক প্রসেসের স্টেটমেন্ট সম্পাদিত হয়, কখনও অন্য প্রসেসের স্টেটমেন্ট সম্পাদিত হয়, আবার কখনও একই প্রসেসের স্টেটমেন্টসমূহ পরপর সম্পাদিত হয়। তবে প্রয়োজনে শর্ত সাপেক্ষে কোন প্রসেসের নির্বাহ নির্দিষ্ট সময় পযর্ন্ত স্থগিত রাখা যায় কিংবা বন্ধ করে দেয়া যায়। নিম্নে একটি মাল্টিথ্রেড প্রোগ্রামের ব্লক চিত্র দেয়া হল।



সিনটেক্স (Syntax) ঃ মূলত run () মেথড ব্যবহার করে থ্রেড বাস্তবায়ন করা হয়। run () মেথডই থ্রেডের সকল কার্যক্রমের কেন্দ্রবিন্দু। এটি ব্যবহার করেই থ্রেডকে ডিফাইন করা হয় এবং থ্রেডের সকল কার্যক্রম বাস্তবায়ন করা হয়। সংশ্লিষ্ট থ্রেডের যে কোন অবজেক্ট সহযোগে run ( ) মেথডটি কল (Call) করা যায়। একে সাধারণত পাবলিক মেথড হিসেবে ডিক্লেয়ার করা হয় যেন অন্য যে কোন প্যাকেজে একে ব্যবহার করা যায়। এর সিনটেক্স (Syntax) নিম্নরূপ–

```
public void run ()
{
// body of the thread
}
```

১০.২ প্রসেস বেইজড মাল্টিটাস্কিং ও থ্রেড বেইজড মাল্টিটাস্কিং এর মধ্যে পার্থক্য (Difference between Process Based and Tread Based Multitasking) ঃ

যে প্রক্রিয়ায় একই সময়ে একই সাথে একাধিক কাজ (task) সম্পন্ন করা যায় তাকে মাল্টিটাস্কিং (Multitasking) বলে। সাধারণত কম্পিউটার এপ্লিকেশনে দুই ধরনের মাল্টিটাস্কিং বিদ্যমান। যথা ঃ

- প্রসেস বেইজড মাল্টিটাস্কিং (Process based multitasking) ও
- থ্রেড বেইজড মাল্টিটাস্কিং (Thread based multitasking)

যদিও এ দুটো পদ্ধতি প্রায় কাছাকাছি। তবুও এদের মধ্যে বেশকিছু পার্থক্য বিদ্যমান। নিম্নে প্রসেস বেইজড মাল্টিটাস্কিং ও থ্রেড বেইজ মাল্টিটাস্কিং পদ্ধতির পার্থক্য উল্লেখ করা হল–

প্রসেস বেইজড মাল্টিটাস্কিং	থ্রেড বেইজড মাল্টিটাস্কিং
একই সময়ে একই সাথে ভিন্ন ভিন্ন অ্যাপ্লিকেশনের ভিন্ন ভিন্ন কাজ (প্রসেস) সম্পন্ন করার প্রক্রিয়াকে প্রসেস বেইজড মাল্টিটাস্কিং বলে।	একই সময়ে একই সাথে একই অ্যাপ্লিকেশনের ভিন্ন ভিন্ন কাজ (প্রসেস) সম্পন্ন করার প্রক্রিয়াকে থ্রেড বেইজড মাল্টিটাস্কিং বলে।
উদাহরণ ঃ একই সময়ে একই সাথে Ms-word এ কোন ডকুমেন্টে কাজ করা, মিডিয়া প্লেয়ারে গান শোনা।	**উদাহরণ ঃ** একই সময়ে একই সাথে Ms-word এ কোন ডকুমেন্ট টাইপ করা, ডকুমেন্ট প্রিন্ট করা।
মাল্টিটাস্কিং পদ্ধতিতে প্রোগ্রাম হচ্ছে একটি কোড-এর সর্বনিম্ন ইউনিট যা শিডিউলার (Scheduler) কর্তৃক পৃথকভাবে ব্যবহৃত হতে পারে।	মাল্টিটাস্কিং পদ্ধতিতে থ্রেড হচ্ছে একটি কোড এর সর্বনিম্ন ইউনিট যা পৃথকভাবে প্রোগ্রামার ব্যবহার করতে পারে।
এ পদ্ধতিতে থ্রেড বেইজড মাল্টিটাস্কিং এর তুলনায় বেশি ওভারহেড (Overhead) প্রয়োজন হয়।	এ পদ্ধতিতে প্রসেস বেইজড মাল্টিটাস্কিং পদ্ধতির তুলনায় কম ওভারহেড (Overhead) প্রয়োজন হয়।
এ পদ্ধতিতে বেশি পরিমাণ টাস্ক (task) সম্পন্ন প্রসেসসমূহের পৃথক পৃথক অ্যাড্রেস স্পেস (Address space) প্রয়োজন হয়।	এ পদ্ধতিতে প্রসেসসমূহ একই (same) অ্যাড্রেস স্পেস (Address space) শেয়ার করে।
এ পদ্ধতিতে প্রসেসসমূহের পারস্পরিক যোগাযোগ ব্যবস্থা ব্যয়বহুল এবং সীমিত।	থ্রেড বেইজড মাল্টিটাস্কিং পদ্ধতিতে প্রসেসসমূহের পাস্পরিক যোগাযোগব্যবস্থা অপেক্ষাকৃত কম ব্যয়বহুল।
যখন কোন জাভা প্রোগ্রাম প্রসেস বেইজড মাল্টিটাস্কিং পরিবেশে কাজ করে তখন এ পদ্ধতিটি জাভা প্রোগ্রামের নিয়ন্ত্রণাধীন হিসেবে কাজ করে না।	মাল্টিথ্রেডেড মাল্টিটাস্কিং পদ্ধতিটি জাভার নিয়ন্ত্রণাধীন হিসেবে কাজ করে।
প্রসেস বেইজড মাল্টিটাস্কিং পদ্ধতিতে আউডল টাইম (Idle time) সীমিত।	থ্রেড বেইজড মাল্টিটাস্কিং পদ্ধতিতে অপেক্ষাকৃত আইডল টাইম বেশি ব্যবহৃত হয়।
প্রসেস বেইজড মাল্টিটাস্কিং পদ্ধতি অপেক্ষাকৃত দ্রুতগতি সম্পন্ন।	থ্রেড বেইজড মাল্টিটাস্কিং পদ্ধতি অপেক্ষাকৃত ধীরগতি সম্পন্ন।

১০.৩ স্টেট ডায়াগ্রামসহ থ্রেড ক্লাসের বিভিন্ন মেথডসমূহ (Several Methods or Thread Class with State Diagram) ঃ

থ্রেড স্টেট (Thread state) ঃ কোন একটি থ্রেডের ক্ষেত্রে বেশ কয়েক ধরনের স্টেট (state) লক্ষ করা যায়। স্টেটসমূহ হচ্ছে–

- বর্ণ (Born) স্টেট
- রেডি (Ready) স্টেট
- রানিং (Running) স্টেট
- ওয়েটিং (Waiting) স্টেট

উদাহরণ ঃ

```
class MyThread extends Thread
{
public void run( )
{
for (int i=1; i<=5; i=i+2)
System.out.println("Inside Thread A");
}
}
class ExampleThread                    // Main Thread
{
public static void main(String args[])
{
MyThread Th=new MyThread();
System.out.println(" Starting MyThread");
Th.start( );                           // Calls run( ) Method
System.out.println("Exit from Main Thread");
}
}
```

উপরোক্ত প্রোগ্রামে 'Thread' ক্লাসকে extend করে 'Mythread' নামে একটি ক্লাস ডিফাইন করা হয়েছে। ক্লাস বডিতে run() মেথড ডিফাইন করা হয়েছে।

implement পদ্ধতি ঃ এই পদ্ধতিতে Runable ইন্টারফেসকে বাস্তবায়ন করে একটি ক্লাস ডিফাইন করা হয় যাতে run( ) মেথডটি বর্ণনার মাধ্যমে ওভাররাইড করা হয়। অতঃপর main( ) মেথড ধারণকারী মেইন থ্রেডে বর্ণিত থ্রেড ক্লাসের একটি অবজেক্ট ডিক্লেয়ার করা হয় যা Thread ক্লাসের অবজেক্ট ডিক্লারেশনের জন্য সংশ্লিষ্ট কনস্ট্রাক্টর মেথডের আরগুমেন্ট হিসেবে ব্যবহৃত হয়। সবশেষে Thread ক্লাসের অবজেক্টের মাধ্যমে তার start( ) মেথড কল করে পরোক্ষভাবে থ্রেড ক্লাসে বর্ণিত run( ) মেথড কল করা হয়।

ইহার ফরম্যাট হলঃ

```
class MyThread implements Runable
{
// Body of the Thread
}
```

উদাহরণ ঃ

```
class A implements Runnable          //Thread A
{
public void run ( )
{
for (int i=1; i<=5; i=i+2)
{
System.out.println("Inside Thread A : i= "+i);
}
System.out.println("Exit from A");
}
}
class B implements Runnable          //Thread B
{
public void run( )
{
for (int j=2; j<=5; j=j+2)
{
System.out.println("Inside Thread B : i= "+j);
}
System.out.println("Exit from B");
}
}
class ExampleImple                   //Main Thread
{
public static void main(String args[])
{
A objA=new A( );                     //Creating Object of Thread A
Thread Th1=new Thread(objA);
B objB=new B( );
Thread Th2=new Thread(objB);
System.out.println("Starting Thread A:");
Th1.start( );                        //starting Thread A
System.out.println("Starting Thread B:");
Th2.start( );                        //Starting Thread B
System.out.println("Exit From Main Thread");
}
}
```

১০.৫ থ্রেডসমূহের অগ্রগণ্যতা (Thread Priorities) ঃ

জাভা প্রোগ্রামের থ্রেড গ্রুপে ব্যবহৃত প্রতিটি থ্রেডের জন্য একটি নির্দিষ্ট অগ্রগণ্যতা (Priority) পরিলক্ষিত হয়। সাধারণত কম অগ্রগণ্যতা বিশিষ্ট থ্রেড অপেক্ষা অধিক অগ্রগণ্যতা বিশিষ্ট থ্রেডসমূহ আগে সম্পাদিত হয়। সুতরাং থ্রেডসমূহের ভিন্ন ভিন্ন অগ্রগণ্যতা নির্ধারণপূর্বক সেগুলো আগে পরে এক্সিকিউট করা যায়। থ্রেডসমূহের অগ্রগণ্যতা নির্ধারণের জন্য Thread ক্লাসের একটি বিশেষ কন্ট্রোল মেথড setPriority ( ) ব্যবহৃত হয়।

setPriority ( ) মেথডের ফরম্যাট হল ঃ

ThreadName.setPriority(PriorityNumber);

1	MIN_PRIORITY	সর্বনিম্ন (Minimum)
5	NORM_PRIORITY	ডিফল্ট (Default)
10	MAX_PRIORITY	সর্বোচ্চ (Maximum)

যেখানে PriorityNumber একটি পূর্ণ সংখ্যা যার মান ১ থেকে ১৫ এর মধ্যবর্তী যে কোন সংখ্যা হতে পারে। স্বাভাবিকভাবে সকল থ্রেডের সমান অগ্রগণ্যতা নির্ধারিত হয়। অবশ্য setPriority মেথড আরগুমেন্ট হিসেবে পূর্ণ সংখ্যা ব্যবহারের পরিবর্তে MIN_PRIORITY, NORM_PRIORITY ও MAX_PRIORITY কন্সট্যান্ট মান ব্যবহৃত হয়, যাদের মান ১, ৫ ও ১০ হিসেবে নির্ধারিত হয়।

আবার getPriority( )মেথড ব্যবহার করে কোন থ্রেডের অগ্রগণ্যতা নির্ণয় করা যায়।

getPriority( ) মেথড ব্যবহারের ফরম্যাট হলঃ

ThreadName.getPriority();

আবার প্রয়োজনে getPriority( ) ও setPriority( ) মেথডের সাথে পূর্ণ সংখ্যা যোগ/বিয়োগ করা যায়। তবে সবক্ষেত্রে লক্ষ্য রাখতে হয় যেন, কোনভাবেই থ্রেডের অগ্রগণ্যতা ১ থেকে ১৫ সীমার বাইরে না হয়।

নিম্নে getPriority( ) ও setPriority( ) মেথড ব্যবহার করে একটি প্রোগ্রাম দেয়া হল।

```
class A extends Thread          //Thread A
{
public void run( )
{
for (int i=1; i<=5; i=i+2)
{
System.out.println("Inside Thread A : i="+i);
}
System.out.println("Exit From A");
}
}
class B extends Thread          //Thread B
{
```

```
public void run( )
{
for (int j=1; j<=5; j=j+2)
{
System.out.println("Inside Thread B : j="+j);
}
System.out.println("Exit From B");
}
}
class C extends Thread          //Thread C
{
public void run( )
{
for (int k=1; k<=5; k=k+2)
{
System.out.println("Inside Thread C : k="+k);
}
System.out.println("Exit From C");
}
}
class ExampleThreads
{
public static void main(String args[])
{
A Th1=new A();
B Th2=new B();
C Th3=new C();
Th1.setPriority(Thread.MIN_PRIORITY);
Th2.setPriority(Thread.MAX_PRIORITY);
Th3.setPriority(Th1.getPriority()+2);
System.out.println("Starting Thread A");
Th1.start( );
System.out.println("Starting Thread B");
Th2.start( );
System.out.println("Starting Thread C");
Th3.start( );
System.out.println("Exit From Main Thread");
}
}
```

১০.৬ থ্রেড সিনক্রোনাইজেশন ও ইন্টারথ্রেড কমিউনিকেশন (Thread Synchronization and Interthread Communication) ঃ

থ্রেড সিনক্রোনাইজেশন (Thread synchronization) ঃ রান (run) মেথড ব্যবহার করে থ্রেড বাস্তবায়নের ফলে যেমন কোন থ্রেডের জন্য তার ডাটা ও মেথডসমূহ বাহির হতে ব্যবহার করা যায় ঠিক তেমনি দুই বা ততোধিক থ্রেড ও একই মেথড শেয়ার করতে পারে। তবে এক্ষেত্রে সমস্যা হল যে, একাধিক থ্রেড একই সময়ে কোন মেথড শেয়ার করতে চাইলে প্রোগ্রামে অচলাবস্থা (Deadlock) সৃষ্টি হতে পারে কিংবা কাঙ্ক্ষিত ফলাফল নাও পাওয়া যেতে পারে। তাই যে প্রক্রিয়ায় প্রোগ্রামকে ডেডলক মুক্ত রেখে দুই বা ততোধিক থ্রেড একই মেথডকে শেয়ার করতে পারে তাকে থ্রেড সিনক্রোনাইজেশন (Thread synchronization) বলে।

জাভা ভাষায় মনিটর (Monitor)নামে এক বিশেষ প্রক্রিয়ায় থ্রেড সিনক্রোনাইজেশন প্রক্রিয়া বাস্তবায়ন করা হয়। মনিটর হল একটি অবজেক্ট বিশেষ যা কোন থ্রেডকে block করে দিতে পারে ও প্রয়োজনে থ্রেডকে revive করতে পারে। যে কোন সময়ে কেবল একটি এবং একটি মাত্র থ্রেড এই মনিটর অবজেক্ট ধারণ করতে পারে। যখন কোন থ্রেড এই লক ধারণ করে তখন তা মনিটরে ধারণ করা বুঝায়। আর এই ধারক থ্রেড সেই মনিটর পরিত্যাগ না করা পর্যন্ত অন্য সকল থ্রেডের (যেগুলো এই মনিটর লক ধারণ করতে চায়) নির্বাহ স্থগিত হয়ে যায়। প্রয়োজনে ধারক থ্রেড পরবর্তি সময়ে এই মনিটরে ধারণ করতে পারে। ফলে সকল রিসোর্স যে কোন সময়ে কেবল একটি মাত্র থ্রেড কর্তৃক ব্যবহৃত হয়। তবে নিম্নোক্ত দুটি উপায়ে থ্রেড সিনক্রোনাইজেশন পদ্ধতি বাস্তবায়ন করা হয়ঃ

- সিনক্রোনাইজড মেথড ব্যবহার করে।
- সিনক্রোনাইজড স্টেটমেন্ট ব্যবহার করে।

সিনক্রোনাইজড মেথড ব্যবহার পদ্ধতি (Synchronized method) ঃ

এই পদ্ধতিতে একাধিক থ্রেড কর্তৃক ব্যবহারযোগ্য মেথডকে সিনক্রোনাইজড মেথড হিসেবে ঘোষণা করা হয়। কোন মেথডকে সিনক্রোনাইজড হিসেবে ঘোষণার জন্য তার রিটার্ন টাইপের পূর্বে অতিরিক্ত synchronized কীওয়ার্ড ব্যবহৃত হয়।

ইহার ফরম্যাট হলঃ

```
synchronized void MethodName( )
{
// Method Body
}
```

সিনক্রোনাইজড স্টেটমেন্ট ব্যবহার পদ্ধতি (Synchronized Statements) ঃ সিনক্রোনাইজড মেথড পদ্ধতির কিছু সীমাবদ্ধতা আছে। যেমন ঃ যে ক্লাসের অবজেক্টকে সিনক্রোনাইজড করতে হবে সেটি মাল্টিথ্রেড এক্সেসিং-এর উপযোগী নয় কিংবা সেটির সোর্স কোড পরিবর্তন যোগ্য নয়। সেক্ষেত্রে এরূপ অবজেক্টকে স্টেটমেন্ট ব্লকের মধ্যে ব্যবহার করা হয়।

ইহা ব্যবহারের ফরম্যাট হল ঃ

```
synchronized (ObjectName)
{
// Statements to be Synchronized
}
```

এখানে, ObjectName হল যে ক্লাসকে সিনক্রোনাইজড করতে হবে সেই ক্লাসের কোন অবজেক্ট। আর যদি কেবল একটি স্টেটমেন্টকে সিনক্রোনাইজড করতে হয় তবে দ্বিতীয় বন্ধনীর ব্যবহার আবশ্যক নয়।

নিম্নে থ্রেড সিক্রোনাইজেশন প্রক্রিয়া বাস্তবায়নের উদাহরণ হিসেবে একটি নমুনা প্রোগ্রাম দেয়া হলঃ

```
class Global                          //Global Class
{
synchronized void CallMe(String Msg)  //sychronized Method
{
System.out.print("["+Msg);
try {
Thread.sleep(1000);
}
catch(Exception E)
{
System.out.print("Exception Caught.");
}
System.out.println("]");
}
}
class Caller implements Runnable      //Caller Thread
{
String Msg;
Global Target;
Thread Th;
public Caller(Global Gb, String S)
{
Target =Gb;
Msg=S;
Th=new Thread(this);
Th.start();
}
public void run()
{
Target.CallMe(Msg);
}
}
class Synchro
{
public static void main(String args[])
```

```
{
Global Obj=new Global();
Caller Obj1=new Caller(Obj, "Hello");
Caller Obj2=new Caller(Obj, "Java");
Caller Obj3=new Caller(Obj, "Programmer");
try{
Obj1.Th.join();
Obj2.Th.join();
Obj3.Th.join();
}
catch(Exception E)
{
System.out.println("Exception Caught.");
}
}
}
```

ইন্টারথ্রেড কমিউনিকেশন (Interthread communication) ঃ যে প্রক্রিয়ায় দুই বা ততোধিক থ্রেডের মধ্যে আন্তঃযোগাযোগ প্রতিষ্ঠিত হয় তাকে ইন্টারথ্রেড কমিউনিকেশন (Interthread communication) বলে। ইন্টারথ্রেড কমিউনিকেশন সিস্টেমে একটি বড় সমস্যা হল পুলিং (Poling)। পুলিং প্রক্রিয়ায় বারবার কিছু শর্ত পরীক্ষণের জন্য একটি লুপ (Loop) বাস্তবায়ন করা হয়। এতে শর্তের মান সত্য হলেই কেবল যথাযথ ব্যবস্থা নেয়া যায়। এতে প্রসেসরের সময় অপচয় রোধ হয়।

জাভা ভাষায় ইন্টারথ্রেড কমিউনিকেশন প্রক্রিয়ায় পুলিং পদ্ধতি রোধকল্পে java.lang প্যাকেজের অন্তর্গত Object ক্লাসের wait( ), notify( ) ও notifyAll( ) মেথড ব্যবহৃত হয়। এসব মেথড কেবল সিনক্রোনাইজড মেথডের মধ্যে ব্যবহৃত হয়। উল্লেখ্য, java.lang প্যাকেজের অন্তর্গত Object ক্লাসটি একটি সার্বজনীন ক্লাস যা স্বয়ংক্রিয়ভাবে সকল ক্লাসে ইনহেরিট হয়। এজন্য তা extend বা implement করতে হয় না।

মেথড	ব্যবহার
wait( )	ব্যবহৃত থ্রেডকে মনিটর অবজেক্ট প্রদান করে অন্য কোন থ্রেড কর্তৃক এই মনিটর অবজেক্ট প্রাপ্ত হলে তা ত্যাগ না করা পর্যন্ত থ্রেডের নির্বাহ স্থগিত রাখে। এটি notify( ) মেথড কল করে।
notify( )	ব্যবহৃত অবজেক্ট wait( ) মেথড ব্যবহারকারী প্রথম থ্রেডকে নির্বাহ তালিকায় আনয়ন করে।
notifyAll( )	ব্যবহৃত অবজেক্ট wait( ) মেথড ব্যবহারকারী সকল থ্রেড নির্বাহ তালিকায় আনয়ন করে। তারমধ্যে থেকে অধিক অগ্রগণ্যতা বিশিষ্ট থ্রেড প্রথমে নির্বাচিত হয়।

১০.৭ থ্রেড সাসপেন্ডিং, রিসিউমিং ও স্টপিং (Suspending, resuming and stopping threads) ঃ

থ্রেড সাসপেন্ডিং (Thread suspending) ঃ থ্রেড সাসপেন্ডিং হল কোন থ্রেড এর কার্যক্রমকে পুরোপুরি অচল না করে অনির্দিষ্ট সময়ের জন্য স্থগিত করা দেয়া। থ্রেড সাসপেন্ড করার জন্য suspend () মেথড ব্যবহার করা হয়। তবে থ্রেড সাসপেন্ড করলে CPU তে ডেডলক অবস্থার সৃষ্টি হতে পারে।

থ্রেড রিসিউমিং (Thread Resuming) ঃ সাসপেন্ডকৃত কোন থ্রেডকে আবার সচল করার প্রক্রিয়াকে থ্রেড রিসিউমিং (Thread Resuming) বলে। থ্রেড রিসিউমিং এর জন্য resume () মেথড ব্যবহার করা হয়।

থ্রেড স্টপিং (Thread stopping) ঃ চলমান কোন থ্রেডের কার্যক্রমকে বন্ধ করার প্রক্রিয়াকে থ্রেড স্টপিং (Thread stopping) বলে। থ্রেডের কার্যক্রমকে বন্ধ করার জন্য stop () মেথড ব্যবহার করা হয়।

১০.৮ মাল্টিথ্রেড প্রোগ্রামিং মেথড ব্যবহার করা প্রোগ্রাম (Programs using multithread programming) ঃ

মাল্টিথ্রেড (Multithread) ঃ মাল্টিথ্রেডেড প্রোগ্রামে একের অধিক থ্রেড ক্লাস বর্ণনা করা হয়। নিম্নে একটি মাল্টিথ্রেডেড প্রোগ্রামের উদাহরণ দেয়া হল ঃ

```
class A extends Thread          //Thread A
{
public void run( )
{
for ( int i=1; i<=5; i=i+2)
{
System.out.println("Inside Thread A: i="+i);
}
System.out.println("Exit from A");
}
}
class B extends Thread          //Thread B
{
public void run( )
{
for ( int j=1; j<=5; j=j+2)
{
System.out.println("Inside Thread B: j="+j);
}
System.out.println("Exit from B");
}
}
```

```
class MultiThread                    //Main Thread
{
public static void main(String args[])
{
A Th1=new A();                       //Creating Object of Thread A
B Th2=new B();                       //Creating Object of Thread B
System.out.println("Starting Thread A");
Th1.start( );
System.out.println("Starting Thread B");
Th2.start( );
System.out.println("Exit from the Main Thread ");
}
}
```

উপরোক্ত প্রোগ্রামে মোট তিনটি আলাদা থ্রেড ক্লাস বর্ণনা করা হয়েছে। যেগুলো যথাক্রমে A , B এবং main( ) মেথড ধারণকারী মেইনথ্রেড ক্লাস Multithread। মেইনথ্রেডের মধ্যে সংশ্লিষ্ট অবজেক্টের মাধ্যমে A , B এবং থ্রেডের start( ) মেথড (পরোক্ষভাবে run( ) মেথড) কল করা হয়েছে। যে মুহূর্তে A থ্রেডের start( ) মেথডের নির্বাহ শুরু হয়, সে মুহূর্তে মোট দুইটি থ্রেড সক্রিয় থাকে, একটি মেইন থ্রেড এবং অন্যটি A থ্রেড। A থ্রেডের run( ) মেথডের নির্বাহ শুরু হওয়া মাত্র start( ) মেথড মেইন থ্রেডে ফিরে আসে এবং B থ্রেডে start( ) মেথডের নির্বাহ শুরু হয়। এতক্ষণে প্রোগ্রামে মোট তিনটি থ্রেড সক্রিয় থাকে, একটি মেইন থ্রেড এবং অন্য দুইটি A ও B থ্রেড। B থ্রেডের run( ) মেথডের নির্বাহ শুরু হওয়া মাত্র start( ) মেথড পুনরায় অপর কোন থ্রেডে ফিরে যায় এবং সেখানকার run( ) মেথড কল করে।

উল্লেখ্য, উপরোক্ত প্রোগ্রামে ব্যবহৃত থ্রেডসমূহের প্রত্যেকটির অগ্রগণ্যতা (Priority) সমান হওয়ায় সেগুলো বিভিন্ন ক্রমে নির্বাহিত হয়। সুতরাং প্রোগ্রাম নির্বাহে উপরোক্ত ফলাফল কিংবা একই ফলাফল নাও পাওয়া যেতে পারে। মূলত থ্রেডগুলোর নির্বাহের কোন ক্রম নিদিষ্ট নাই। তবে থ্রেডগুলো নিয়ন্ত্রণের অবশ্যই সুনিদিষ্ট নিয়ম আছে।

অনুশীলনী-১০

অতি সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। থ্রেড (Thread) কী?

উত্তরঃ একই সময়ে এক সাথে একাধিক কাজ করার প্রক্রিয়াকে থ্রেড (Thread) বলে। যেমন- মিডিয়া প্লেয়ারে গান শোনা, এমএসওয়ার্ডে ডকুমেন্ট কম্পোজ করা, ও প্রিন্টারে প্রিন্ট করা, এই তিনটি কাজ একই সময়ে এক সাথে করার প্রক্রিয়াই হচ্ছে থ্রেড।

২। সিঙ্গেল থ্রেডেড প্রোগ্রাম কাকে বলে?

উত্তরঃ যে কোন জাভা প্রোগ্রামে এক বা একাধিক থ্রেড থাকে। যদি কোন প্রোগ্রামে একটি মাত্র থ্রেড থাকে তবে তাকে সিঙ্গেল থ্রেডেড প্রোগ্রাম (Single threaded program) বলে। main () মেথড বিশিষ্ট প্রোগ্রামসমূহ সিঙ্গেল থ্রেডেড প্রোগ্রামের অন্তর্ভুক্ত। এরূপ প্রোগ্রামের স্টেটমেন্টসমূহ পর্যায়ক্রমে একটির পর একটি সম্পাদিত হয়। সিঙ্গেল থ্রেডেড প্রোগ্রামে কেবল একটি মাত্র কন্ট্রোল ফ্লো (Control flow) থাকে।

৩। জাভা থ্রেডগুলোর অবস্থান কোথায়?

উত্তরঃ মূলত তিনটি স্থানে জাভার থ্রেডের বিষয়গুলো অবস্থান করে। যথা ঃ

- Java. lang. Thread ক্লাসে
- Java. lang. object ক্লাসে এবং
- জাভা ভার্চুয়াল মেশিন (JVM) এ।

৪। থ্রেড Priority বলতে কী বোঝায়?

উত্তরঃ জাভা প্রোগ্রামের থ্রেড গ্রুপে ব্যবহৃত প্রতিটি থ্রেডের জন্য একটি অগ্রগণ্যতা ব্যবহার করা হয়, একে থ্রেড Priority বলা হয়।

থ্রেড Priority তিন প্রকার, যথা-

MIN-PRIORITY, NORM-PRIORITY ও MAX-PRIORITY.

৫। Multithread Programming বলতে কী বুঝায়?

উত্তরঃ একই সাথে একই সিস্টেমের অধীনে একাধিক প্রসেস বা সাব প্রোগ্রাম পরিচালনা করার পদ্ধতিকে মাল্টি থ্রেড প্রোগ্রামিং বলে। এ প্রক্রিয়ায় একই সময়ে একাধিক থ্রেড সক্রিয় থেকে প্রোগ্রামে একাধিক কন্ট্রোল ফ্লোতে কার্যকর হয়, ফলে প্রোগ্রামে কাছাকাছি সময়ে কখনও এক প্রসেসের স্টেটমেন্ট সম্পাদিত হয় কখনও জন্য প্রসেসের স্টেটমেন্ট সম্পাদিত হয়।

যেমন- মাইক্রোসফট ওয়ার্ডে একটি ডকুমেন্ট প্রিন্ট করার পাশাপাশি অপর একটি ডকুমেন্টে লেখা যায়।

৬। Multitasking কী?

উত্তরঃ একই সাথে ভিন্ন ভিন্ন সিস্টেমের প্রোগ্রাম পরিচালনার পদ্ধতিকে মাল্টিটাস্কিং প্রোগ্রামিং বলে। যেমন- একটি ওয়েভ পেজে ব্রাউজিং এর পাশাপাশি অপর এক বা একাধিক ওয়েভ পেজ সার্চিং করা যায়।

৭। Thread synchronization বলতে কী বুঝায়?

উত্তরঃ যে প্রক্রিয়ায় প্রোগ্রামকে অচলাবস্থা (Deadlock) হতে নিরাপদ রেখে দুই বা ততোধিক থ্রেডের জন্য একই মেথড তথা রিসোর্সের ব্যবহার নিশ্চিত করা হয় তাকে থ্রেড সিনক্রোনাইজেশন বলে।

৮। থ্রেড ক্লাস তৈরির প্রক্রিয়াগুলো কী কী? [বাকাশিবো-২০১১, ২০১০]

উত্তরঃ সাধারণত দুইটি উপায়ে থ্রেড ক্লাস তৈরি করা যায় ঃ

- সরাসরি Thread class extend করে এবং
- Runnable (রানেবল) ক্লাসকে implement করে।

৯। wait ( ) মেথডের ব্যবহার লেখ।

উত্তরঃ wait ( ) method ব্যবহৃত থ্রেডকে মনিটর অবজেক্ট প্রদান করে অন্য কোন থ্রেড কর্তৃক এ মনিটর অবজেক্ট প্রাপ্ত হলে তা ত্যাগ না করা পর্যন্ত থ্রেডের নির্বাহ স্থগিত রাখে।

১০। Blocked state কী?

উত্তরঃ যখন কোন থ্রেডকে runnable state অর্থাৎ running state এ উপনীত হওয়া থেকে রোধ করা হয় তখন তাকে Blocked state বলে।

১১। থ্রেড বাস্তবায়ন করা হয় কিভাবে?

উত্তরঃ মূলত run () মেথড ব্যবহার করে থ্রেড বাস্তবায়ন করা হয়। run () মেথডই থ্রেডের সকল কার্যক্রমের কেন্দ্রবিন্দু। এটি ব্যবহার করেই থ্রেডকে ডিফাইন করা হয় এবং থ্রেডের সকল কার্যক্রম বাস্তবায়ন করা হয়। সংশ্লিষ্ট থ্রেডের যে কোন অবজেক্ট সহযোগে run ( ) মেথডটি কল (Call) করা যায়। একে সাধারণত পাবলিক মেথড হিসেবে ডিক্লেয়ার করা হয় যেন অন্য যে কোন প্যাকেজে একে ব্যবহার করা যায়। এর সিনটেক্স (Syntax) নিম্নরূপ–

```
public void run ()
{
// body of the thread
}
```

১২। থ্রেড স্টেট (Thread state) সমূহের নাম উল্লেখ কর।

উত্তরঃ কোন একটি থ্রেডের ক্ষেত্রে বেশ কয়েক ধরনের স্টেট (state) লক্ষ করা যায়। স্টেটসমূহ হচ্ছে–

- বর্ণ (Born) স্টেট
- রেডি (Ready) স্টেট
- রানিং (Running) স্টেট
- ওয়েটিং (Waiting) স্টেট
- স্লিপিং (Sleeping) স্টেট
- ডেড (Dead) স্টেট ও
- ব্লকড (Blocked) স্টেট।

১৩। ইন্টারথ্রেড কমিউনিকেশন কাকে বলে?

উত্তরঃ যে প্রক্রিয়ায় দুই বা ততোধিক থ্রেডের মধ্যে আন্তঃযোগাযোগ প্রতিষ্ঠিত হয় তাকে ইন্টারথ্রেড কমিউনিকেশন (Interthread communication) বলে।

১৪। start ( ) মেথড ও stop ( ) মেথডের কাজ কী?

উত্তরঃ কোন থ্রেডে start ( ) মেথড ব্যবহার করে থ্রেড ক্লাসের run ( ) মেথড নির্বাহ শুরু হয় এবং stop ( ) মেথডের মাধ্যমে চলমান থ্রেডের নির্বাহ বন্ধ করা হয়।

১৫। থ্রেড সাসপেন্ডিং বলতে কী বুঝায়?

উত্তরঃ থ্রেড সাসপেন্ডিং হল কোন থ্রেড এর কার্যক্রমকে পুরোপুরি অচল না করে অনির্দিষ্ট সময়ের জন্য স্থগিত করে দেয়া। থ্রেড সাসপেন্ড করার জন্য suspend () মেথড ব্যবহার করা হয়। তবে থ্রেড সাসপেন্ড করলে CPU তে ডেডলক অবস্থার সৃষ্টি হতে পারে।

১৬। থ্রেড রিসিউমিং কী?

উত্তরঃ সাসপেন্ডকৃত কোন থ্রেডকে আবার সচল করার প্রক্রিয়াকে থ্রেড রিসিউমিং (Thread Resuming) বলে। থ্রেড রিসিউমিং এর জন্য resume () মেথড ব্যবহার করা হয়।

সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। **প্রসেস বেসড ও থ্রেড বেসড মাল্টি টাস্কিং পদ্ধতির পার্থক্য লেখ।** [বাকাশিবো–২০১১, ২০১০]
উত্তর সংকেত ঃ ১০.২ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **যেকোন পাঁচটি থ্রেড কন্ট্রোল মেথডের নাম ও তাদের ব্যবহার লেখ।**
উত্তর সংকেত ঃ ১০.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **উদাহরণসহ Thread's priority ব্যাখ্যা কর।**
উত্তর সংকেত ঃ ১০.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। **আন্তঃথ্রেড কমিউনিকেশন পদ্ধতি ব্যাখ্যা কর।** [বাকাশিবো-২০১১]
উত্তর সংকেত ঃ ১০.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। **sleep ( ) ও wait ( ) মেথড এর পার্থক্য লেখ।**
উত্তর সংকেত ঃ ১০.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। **notify ( ) ও notify All ( ) মেথড ব্যবহারের পার্থক্য লেখ।**
উত্তর সংকেত ঃ ১০.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

রচনামূলক প্রশ্নোত্তর ঃ

১। **থ্রেডের (thread) জীবন চক্র বর্ণনা কর।** [বাকাশিবো–২০১১]
অথবা, স্টেট ডায়াগ্রামসহ থ্রেড ক্লাসের মেথডসমূহ বর্ণনা কর।
উত্তর সংকেত ঃ ১০.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **থ্রেড কন্ট্রোল মেথডগুলো বর্ণনা কর।**
উত্তর সংকেত ঃ ১০.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **প্রোগ্রামের মাধ্যমে মাল্টিথ্রেড পদ্ধতি ব্যাখ্যা কর।**
উত্তর সংকেত ঃ ১০.৮ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। **getpriority ( ) ও setpriority ( ) মেথড ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।**
উত্তর সংকেত ঃ ১০.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। **থ্রেড Synchronization পদ্ধতি ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।**
উত্তর সংকেত ঃ ১০.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। **উদাহরণসহ থ্রেড তৈরির বিভিন্ন পদ্ধতি বর্ণনা কর।**
উত্তর সংকেত ঃ ১০.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

এনিউমিরেশন, অটোবক্সিং ও অ্যানোটেশনস
Enumeration, Autoboxing and annotations

অধ্যায় ১১

১১.০ ভূমিকা (Introduction) ঃ

JDK5 এমন একটি Feature তৈরি করে যা প্রোগ্রামারদের অনেক দিনের কাম্য ছিল। আর তা হচ্ছে এনিউমিরেশন (Enumeration)। সাধারণভাবে এনিউমিরেশন হচ্ছে কতগুলি কন্সট্যান্ট সমষ্টি। যদিও জাভাতে একইরকম কার্যক্ষমতা সম্পন্ন আরও অনেক Feature রয়েছে; যেমন- ফাইনাল ভেরিয়েবল। এতদসত্ত্বেও অনেক প্রোগ্রামার এনিউমিরেশনের Conceptual বিশুদ্ধতাকে মিস করে। এর কারণ মূলত সব ধরনের ল্যাংগুয়েজে এনিউমিরেশন ব্যবহার করা যায় না। JDK5 এর শুরুর দিকে জাভা ভাষায় এনিউমিরেশন সংযুক্ত হয় এবং এটি বর্তমানে জাভা প্রোগ্রামারদের কাছে অতি সহজলভ্য।

সাধারণভাবে অন্যান্য ল্যাংগুয়েজের এনিউমিরেশনের সাথে জাভা এনিউমিরেশনের মিল রয়েছে। যদিও এই মিল অতি সামান্য। যেমন C++ এ এনিউমিরেশন কতগুলি ইন্টিজার কন্সট্যান্টের সমষ্টি কিন্তু জাভাতে এনিউমিরেশন একটি Class Type নির্ধারণ করে। এনিউমিরেশনের মাধ্যমে Class তৈরির ফলে এনিউমিরেশনের ধারণা ব্যাপকভাবে প্রসারিত হয়েছে। উদাহরণস্বরূপ, জাভাতে একটি এনিউমিরেশনের Constructors, Methods এবং Instance Variable থাকতে পারে। যদিও এনিউমিরেশন তৈরিতে কয়েক বছর লেগেছে তথাপি এটি জাভাকে অনেক বেশি সমৃদ্ধ করেছে।

১১.১ সিনটেক্সসহ এনিউমিরেশন (Enumeration with Syntax) ঃ

এনিউমিরেশন (Enumeration) ঃ এনিউমিরেশন (Enumeration) হচ্ছে কতগুলো কন্সট্যান্ট নামের সমষ্টি। মূলত ইউজার ডিফাইন্ড ডাটা টাইপ তৈরির একটি কৌশল হচ্ছে এনিউমিরেশন।

সিনটেক্স (Syntax) ঃ এনিউমিরেশন তৈরির ফরম্যাট হল-

```
enum days
}
Saturday, Sunday, Monday, Tuesday, Friday
}
```

এখানে 'enum' হল এনিউমিরেশন তৈরির কী ওয়ার্ড। Saturday, Sunday, Monday, Tuesday, Friday ইত্যাদির প্রতিটিই হল এক একটি কন্সট্যান্ট। প্রতিটি কন্সট্যান্ট 'days' ক্লাসের স্ট্যাটিক ফাইনাল (Static final) মেম্বার এবং পাবলিক হিসেবে ডিক্লেয়ার করা হয়েছে।

জাভায় এ কনস্ট্যান্টগুলোকে self typed বলা হয়। কারণ ডিক্লেয়ারকৃত কন্সট্যান্টসমূহ এবং এনিউমিরেশন এর ডাটা টাইপ একই (Same)।

১৫। থ্রেড সাসপেন্ডিং বলতে কী বুঝায়?

উত্তরঃ থ্রেড সাসপেন্ডিং হল কোন থ্রেড এর কার্যক্রমকে পুরোপুরি অচল না করে অনির্দিষ্ট সময়ের জন্য স্থগিত করে দেয়া। থ্রেড সাসপেন্ড করার জন্য suspend () মেথড ব্যবহার করা হয়। তবে থ্রেড সাসপেন্ড করলে CPU তে ডেডলক অবস্থার সৃষ্টি হতে পারে।

১৬। থ্রেড রিসিউমিং কী?

উত্তরঃ সাসপেন্ডকৃত কোন থ্রেডকে আবার সচল করার প্রক্রিয়াকে থ্রেড রিসিউমিং (Thread Resuming) বলে। থ্রেড রিসিউমিং এর জন্য resume () মেথড ব্যবহার করা হয়।

➥ *সংক্ষিপ্ত প্রশ্নোত্তর ঃ*

১। **প্রসেস বেসড ও থ্রেড বেসড মাল্টি টাস্কিং পদ্ধতির পার্থক্য লেখ।** [বাকাশিবো-২০১১, ২০১০]

উত্তর সংকেত ঃ ১০.২ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **যেকোন পাঁচটি থ্রেড কন্ট্রোল মেথডের নাম ও তাদের ব্যবহার লেখ।**

উত্তর সংকেত ঃ ১০.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **উদাহরণসহ Thread's priority ব্যাখ্যা কর।**

উত্তর সংকেত ঃ ১০.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। **আন্তঃথ্রেড কমিউনিকেশন পদ্ধতি ব্যাখ্যা কর।** [বাকাশিবো-২০১১]

উত্তর সংকেত ঃ ১০.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। **sleep ( ) ও wait ( ) মেথড এর পার্থক্য লেখ।**

উত্তর সংকেত ঃ ১০.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। **notify ( ) ও notify All ( ) মেথড ব্যবহারের পার্থক্য লেখ।**

উত্তর সংকেত ঃ ১০.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

➥ *রচনামূলক প্রশ্নোত্তর ঃ*

১। **থ্রেডের (thread) জীবন চক্র বর্ণনা কর।** [বাকাশিবো-২০১১]
অথবা, স্টেট ডায়াগ্রামসহ থ্রেড ক্লাসের মেথডসমূহ বর্ণনা কর।

উত্তর সংকেত ঃ ১০.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **থ্রেড কন্ট্রোল মেথডগুলো বর্ণনা কর।**

উত্তর সংকেত ঃ ১০.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **প্রোগ্রামের মাধ্যমে মাল্টিথ্রেড পদ্ধতি ব্যাখ্যা কর।**

উত্তর সংকেত ঃ ১০.৮ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। **getpriority ( ) ও setpriority ( ) মেথড ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।**

উত্তর সংকেত ঃ ১০.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। **থ্রেড Synchronization পদ্ধতি ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।**

উত্তর সংকেত ঃ ১০.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। **উদাহরণসহ থ্রেড তৈরির বিভিন্ন পদ্ধতি বর্ণনা কর।**

উত্তর সংকেত ঃ ১০.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

এনিউমিরেশন, অটোবক্সিং ও অ্যানোটেশনস
Enumeration, Autoboxing and annotations

অধ্যায় ১১

১১.০ ভূমিকা (Introduction) ঃ

JDK5 এমন একটি Feature তৈরি করে যা প্রোগ্রামারদের অনেক দিনের কাম্য ছিল। আর তা হচ্ছে এনিউমিরেশন (Enumeration)। সাধারণভাবে এনিউমিরেশন হচ্ছে কতগুলি কন্সট্যান্ট সমষ্টি। যদিও জাভাতে একইরকম কার্যক্ষমতা সম্পন্ন আরও অনেক Feature রয়েছে; যেমন– ফাইনাল ভেরিয়েবল। এতদসত্ত্বেও অনেক প্রোগ্রামার এনিউমিরেশনের Conceptual বিশুদ্ধতাকে মিস করে। এর কারণ মূলত সব ধরনের ল্যাংগুয়েজে এনিউমিরেশন ব্যবহার করা যায় না। JDK5 এর শুরুর দিকে জাভা ভাষায় এনিউমিরেশন সংযুক্ত হয় এবং এটি বর্তমানে জাভা প্রোগ্রামারদের কাছে অতি সহজলভ্য।

সাধারণভাবে অন্যান্য ল্যাংগুয়েজের এনিউমিরেশনের সাথে জাভা এনিউমিরেশনের মিল রয়েছে। যদিও এই মিল অতি সামান্য। যেমন C++ এ এনিউমিরেশন কতগুলি ইন্টিজার কন্সট্যান্টের সমষ্টি কিন্তু জাভাতে এনিউমিরেশন একটি Class Type নির্ধারণ করে। এনিউমিরেশনের মাধ্যমে Class তৈরির ফলে এনিউমিরেশনের ধারণা ব্যাপকভাবে প্রসারিত হয়েছে। উদাহরণস্বরূপ, জাভাতে একটি এনিউমিরেশনের Constructors, Methods এবং Instance Variable থাকতে পারে। যদিও এনিউমিরেশন তৈরিতে কয়েক বছর লেগেছে তথাপি এটি জাভাকে অনেক বেশি সমৃদ্ধ করেছে।

১১.১ সিনটেক্সসহ এনিউমিরেশন (Enumeration with Syntax) ঃ

এনিউমিরেশন (Enumeration) ঃ এনিউমিরেশন (Enumeration) হচ্ছে কতগুলো কন্সট্যান্ট নামের সমষ্টি। মূলত ইউজার ডিফাইন্ড ডাটা টাইপ তৈরির একটি কৌশল হচ্ছে এনিউমিরেশন।

সিনটেক্স (Syntax) ঃ এনিউমিরেশন তৈরির ফরম্যাট হল–

```
enum days
}
Saturday, Sunday, Monday, Tuesday, Friday
}
```

এখানে 'enum' হল এনিউমিরেশন তৈরির কী ওয়ার্ড। Saturday, Sunday, Monday, Tuesday, Friday ইত্যাদির প্রতিটিই হল এক একটি কন্সট্যান্ট। প্রতিটি কন্সট্যান্ট 'days' ক্লাসের স্ট্যাটিক ফাইনাল (Static final) মেম্বার এবং পাবলিক হিসেবে ডিক্লেয়ার করা হয়েছে।

জাভায় এ কনস্ট্যান্টগুলোকে self typed বলা হয়। কারণ ডিক্লেয়ারকৃত কন্সট্যান্টসমূহ এবং এনিউমিরেশন এর ডাটা টাইপ একই (Same)।

১১.২ ভ্যালু ও ভ্যালু অব মেথডের মধ্যে পার্থক্য (Difference between Value ( ) and Value of ( ) Methods) ঃ

এনিউমিরেশনে দুই ধরনের বিল্ট-ইন মেথড আছে। যেমন–

– ভ্যালু (Value) মেথড এবং

– ভ্যালু অব (Value of) মেথড।

ভ্যালু মেথড (Values method) ঃ ভ্যালু মেথডটি একটি অ্যারে রির্টান করে যা কতগুলো এনিউমিরেশন কনস্ট্যান্ট এর সমষ্টি। এর ডাটা টাইপ হচ্ছে 'enum'। values( ) মেথডের ফরম্যাট নিম্নরূপ--

public static enum-type [ ] values ( )

ভ্যালু অব মেথড (Value of method) ঃ value of ( ) মেথড একটি স্ট্রিং এনিউমিরেশন কনস্ট্যান্ট এর মান রির্টান করে যা 'str' হিসেবে ব্যবহৃত। value of ( ) মেথডের ফরম্যাট নিম্নরূপ–

public static enum-type [ ] value of (string str)

১১.৩ জাভা বাসায় টাইপ র‍্যাপার (Type Wrappers in Java) ঃ

টাইপ র‍্যাপার (Type wrapper) ঃ জাভা ভাষায় প্রোগ্রামিং এর দক্ষতা বৃদ্ধির জন্য বিভিন্ন ধরনের প্রিমিটিভ বা মৌলিক ডাটা টাইপ (int, char, float, double ইত্যাদি) ব্যবহার করা হয়। কিন্তু প্রিমিটিভ ডাটা টাইপের ক্ষেত্রে সমস্যা হল ঃ মেথডের মাধ্যমে কোন প্রিমিটিভ ডাটা টাইপ pass করা সম্ভব নয়। তাই বিকল্প হিসেবে অবজেক্ট ব্যবহার করা হয়। কিন্তু অবজেক্ট ব্যবহারের ক্ষেত্রেও ক্যালকুলেশনের সময় অনাকাঙ্ক্ষিত ওভারহেড (Overhead) উৎপন্ন হয়। মৌলিক ডাটা টাইপ ও অবজেক্ট ব্যবহারে নানা জটিলতা এড়ানোর জন্য জাভা ভাষায় এক বিশেষ ধরনের ক্লাস ব্যবহার করা হয় যেখানে একটি অবজেক্টের ভিতরে অনেকগুলো মৌলিক ডাটা টাইপকে 'encapsulate' করে অবজেক্ট ও মৌলিক ডাটা টাইপের মধ্যে সমন্বয় সাধন করা হয়। উক্ত বিশেষ ক্লাসকেই টাইপ ব্যাপার (Type wrapper) বলে।

জাভায় প্রধানত দুই ধরনের টাইপ র‍্যাপার (Type wrapper) ব্যবহৃত হয়। যথা ঃ

- নন নিউমেরিক টাইপ র‍্যাপার (Non Numeric Type wrapper) এবং
- নিউমেরিক টাইপ র‍্যাপার (Numeric Type wrapper)।

নন নিউমেরিক টাইপ র‍্যাপার (Non Numeric Type wrapper) ঃ নন নিউমেরিক টাইপ র‍্যাপার দুই ধরনের। যথা ঃ

- ক্যারেক্টার (Character) র‍্যাপার ও
- বুলিয়ান (Boolean) র‍্যাপার।

ক্যারেক্টার র‍্যাপার (Character wrapper) ঃ 'char' কী ওয়ার্ড ব্যবহার করে ক্যারেক্টার র‍্যাপার গঠিত। এর ফরম্যাট হল ঃ

character (char ch)

ক্যারেক্টার অবজেক্ট হতে char value পেতে হলে char value কল করতে হয়। যেমন–

char char value ( )

বুলিয়ান র‍্যাপার (Boolean wrapper) ঃ boolean কীওয়ার্ড ব্যবহার করে বুলিয়ান র‍্যাপার গঠিত। এর ফরম্যাট হল ঃ

```
boolean (boolean boolValue)
boolean (string boolString)
```

প্রথম ক্ষেত্রে, boolValue হয় true হবে অথবা false হবে। দ্বিতীয় ক্ষেত্রে, যদি boolString, "true" string এর মান ধারণ করে (Uppercase ও Lowercase উভয় ক্ষেত্রে), তখন নতুন Boolean object true হবে। অন্যথায় এটি false হবে।

Boolean object থেকে boolean মান পেতে হলে নিম্নরূপ booleanValue( ) ব্যবহার করতে হবেঃ

```
boolean boolValue ( )
```

এটি প্রার্থিত object এর boolean equivalent মান return করে।

নিউমেরিক টাইপ র‍্যাপার (Numeric Type wrapper) ঃ জাভায় ছয় ধরনের নিউমেরিক টাইপ র‍্যাপার রয়েছে। যথা ঃ

- বাইট (byte) টাইপ র‍্যাপার
- ডাবল (double) টাইপ র‍্যাপার
- ফ্লোট (float) টাইপ র‍্যাপার
- ইন্টিজার (int) টাইপ র‍্যাপার
- লং (long) টাইপ র‍্যাপার এবং
- শর্ট (short) টাইপ র‍্যাপার

নিউমেরিক টাইপ র‍্যাপারসমূহ একটি অ্যাবস্ট্রাক্ট ক্লাস নাম্বার এর বৈশিষ্ট্য ধারণ করে। নাম্বার একটি মেথড ডিক্লেয়ার করে যা কোন অবজেক্টের মানকে পৃথক পৃথক নাম্বার ফরম্যাটে প্রদর্শন করে। যেমন–

```
byte byteValue ( )
double doubleValue ( )
float floatValue ( )
int intValue ( )
long longValue ( )
short shortValue ( )
```

১১.৪ বিভিন্ন ধরনের অটোবক্সিং/আন বক্সিং (Different Types of Autoboxing/ Unboxing) ঃ

অটোবক্সিং/আনবক্সিং (Autoboxing/Unboxing) ঃ Autoboxing এবং Auto- Unboxing এমন একটি পদ্ধতি যার মাধ্যমে কোন মৌলিক ডাটা টাইপের সমতুল্য কোন type wrapper এর মধ্যে স্বয়ংক্রিয়ভাবে এ্যানক্যাপসুলেটেড (boxed) অবস্থায় থাকতে পারে এবং যেকোন সময় প্রয়োজন অনুসারে ঐ টাইপের অবজেক্ট ব্যবহৃত হতে পারে।

Autounboxing এমন একটি পদ্ধতি যার মাধ্যমে টাইপ র‍্যাপারের মান প্রয়োজনের সময় স্বয়ংক্রিয়ভাবে টাইপ র‍্যাপার হতে extract করা যায়।

এখানে কোন method যেমন intValue ( ) বা doubleValue( ) ইত্যাদি call করার প্রয়োজন পড়ে না।

প্রকারভেদ (Types) ঃ জাভা মূলত দুই ধরনের টাইপ র‍্যাপারের জন্য অটোবক্সিং/আনবক্সিং (Autoboxing/ Unboxing) সাপোর্ট করে। যথা ঃ

- বুলিয়ান টাইপ র‍্যাপার অটোবক্সিং/আনবক্সিং (Boolean type wrapper autoboxing/ unboxing)
- ক্যারেক্টার টাইপ র‍্যাপার অটো বক্সিং/আন বক্সিং (Character type wrapper autoboxing/ Unboxing)

নিম্নে বুলিয়ান ও ক্যারেক্টার টাইপ র‍্যাপার অটোবক্সিং/আনবক্সিং-এর উদাহরণ উল্লেখ করা হল ঃ

```
class AutoBox {
public static void main (String args [ ]) {
// Autobox / unbox a boolean
boolean b = true;
// Below, b is auto-unboxed when used in
// a conditional expression, such as an if.
if (b) system.out.printIn("b is true");
// Autobox / unbox a char.
character ch = 'x'; // box a char
char ch2 = ch; // unbox a char
// System.out.printIn("ch2 is" +ch2);
```

এই program এর output টি নিম্নরূপ হবে ঃ

```
b is true
ch2 is x
```

উল্লেখিত program এ সবচেয়ে গুরুত্বপূর্ণ বিষয় হচ্ছে, if conditional expression এর ভিতরে b কে auto-unboxing করা হয়েছে। আমরা জানি যে, if conditional expression boolean type কে মূল্যায়ন (evaluate) করে। যখন conditional expression মূল্যায়ন করা হয় তখন auto-unboxing এর কারণে b এর boolean মান স্বয়ংক্রিয়ভাবে unboxed হয়ে যায়। এইভাবে autoboxing/unboxing এর সুবিধা নিয়ে boolean object কে if statement নিয়ন্ত্রণ করার জন্য ব্যবহার করা যায়।

১১.৫ অ্যানোটেশনস ও এর উদাহরণ (Annotations with Examples) ঃ

অ্যানোটেশনস (Annotations) ঃ প্রোগ্রামের মূলভাবকে অপরিবর্তিত রেখে প্রোগ্রামের কোন অ্যাকশনকে পরিবর্তন না করে প্রোগ্রামে বিশেষ করে সোর্স ফাইলে অনেক সময় অতিরিক্ত কিছু ঐচ্ছিক তথ্য বা ইনফরমেশন যুক্ত করা হয়। এসব ঐচ্ছিক অতিরিক্ত তথ্য বা ইনফরমেশনকেই অ্যানোটেশনস (Annotation) বলে। একে মেটা ডাটা (Meta data)ও বলা হয়। এসব ইনফরমেশনগুলো বিভিন্ন ধরনের টুলসের (Tools) ডেভেলপমেন্ট ও বিস্তারের ক্ষেত্রে ব্যবহার করা হয়।

উদাহরণ ঃ

সোর্স কোড (Source code) জেনারেট করার সময় সোর্স ফাইলে সংযুক্ত তথ্যাদিই অ্যানোটেশনস (Annotations)।

সিনটেক্স (Syntax) ঃ ইন্টারফেস (Interface) তৈরিতে অ্যানোটেশনস ডিক্লেয়ার করা যায়। এর ফরম্যাট নিম্নরূপ ঃ

```
@ interface MyAnno
{
String str ( );
int val ( );
}
```

প্রথমত, লক্ষণীয় যে, interface keyword টি @ দিয়ে শুরু হয়েছে। এতে কম্পাইলার বুঝতে পারবে যে অ্যানোটেশনস ডিক্লেয়ার করা হয়েছে। **দ্বিতীয়ত,** এখানে ২টি মেম্বার Str ( ) এবং Val ( ) ব্যবহৃত হয়েছে। প্রতিটি অ্যানোটেশনস (annotation) পৃথকভাবে মেথড ডিক্লারেশন ধারণ করে। যাহোক এই method গুলি Java irnplement করে, আমরা নিজেরা এর body তৈরি করতে পারি না। তদুপরি, এই method গুলি field এর মত কাজ করে।

যখন কোন annotation apply করা হয়, এর memberগুলির মান দিতে হয়। উদাহরণস্বরূপ, এখানে একটি method এ MyAnno ব্যবহার করা হল–

```
@ MyAnno (str = "Annotation Example", val = 100)
public static void MyMeth ( ) { ".....
```

Annotation এর Syntax লক্ষ্য করলে দেখা যায় যে, এর নাম @ দিয়ে শুরু হয়েছে যাকে একটি বন্ধনীর মধ্যস্থ কতগুলি member অনুসরণ করেছে। কোন member কে মান দিতে হলে ঐ member এর মানকে একটি মান দ্বারা assign করতে হয়। এজন্য, উদাহরণটিতে ব্যবহৃত String "Annotation Example" কে MyAnno-এর member str দ্বারা assign করা হয়েছে। লক্ষণীয় যে, এই assignment এ str এ কোন বন্ধনী নাই। যখন কোন annotation member এ মান দেয়া হয় তখন শুধু এর নাম ব্যবহৃত হয়। এভাবে এই উদাহরণটিতে annotation member গুলি field এর মত ব্যবহৃত হয়েছে।

১১.৬ জাভায় ব্যবহৃত বিল্ট ইন অ্যানোটেশনস (Built in Annotations in Java) ঃ

Java-তে বিভিন্ন ধরনের built-in annotation রয়েছে। এর মধ্যে বেশির ভাগই বিশেষ বিশেষ ক্ষেত্রে ব্যবহৃত হয় কিন্তু সাত ধরনের annotation জেনারেল পারপাস হিসাবে ব্যবহৃত হয়। তন্মধ্যে চার ধরনের annotation @ Relation, @ Documented, @ Target, @ Inherited ইত্যাদি java.lang. annotation প্যাকেজ থেকে import করা আর @ Override, @ Deprecated এবং @ Suppress Warnings এই তিন ধরনের annotation java.lang প্যাকেজে সংযুক্ত করা।

নিম্নে এই সাত ধরনের annotation বর্ণনা করা হলঃ

@ Relation ঃ এক annotation থেকে অন্য annotation এ ব্যবহার করার জন্য @ Relation ডিজাইন করা হয়েছে। ইহা relation policy কে স্পেসিফাই করে।

@ Documented ঃ কোন annotation এর ডকুমেন্ট তৈরির জন্য ব্যবহৃত tool এর marker interface হিসাবে @ Documented annotation ব্যবহৃত হয়। এটি annotation declartion এ ব্যবহার করার জন্য ডিজাইন করা হয়েছে।

@ Target ঃ কোন annotation এ ব্যবহৃত ডিক্লারেশনের টাইপ নির্ধারণ করার জন্য @ Target annotation ব্যবহার করা হয়। এটিও অন্য কোন annotation এর annotation হিসাবে ব্যবহার করার জন্য ডিজাইন করা হয়েছে। @ Target একটি argument নেয় যাকে অবশ্যই Element Type enumeration-এর কন্সস্ট্যান্ট হতে হবে। এই argument-ই কোন annotation এর ডিক্লারেশন-এর টাইপ নির্ধারণ করে। নিম্নে ছক আকারে এই ধ্রুবক (constant) গুলির corresponding টাইপ ডিক্লারেশনসহ দেখানো হলঃ

কনস্ট্যান্ট	কমেন্টারি টাইপ ডিক্লারেশন
Target Constant	Annotation Can Be Applied To
ANNOTATION-TAPE	Another annotation
CONSTRUCTOR	Constructor
FIELD	Field
LOCAL_VARIABLE	Local Variable
METHOD	Method
PACKAGE	Package
PARAMETER	Parameter
TYPE	Class, interface or enumeration

@ Inherited ঃ @ Inherited annotation এমন একটি মার্কার annotation যা শুধুমাত্র অন্য annotation ডিক্লারেশন এর ক্ষেত্রে ব্যবহৃত হয়। এছাড়া এটি শুধুমাত্র ক্লাস ডিক্লারেশন এ ব্যবহৃত annotation এর ক্ষেত্রে কাজ করে। @ Inherited annotation সুপার ক্লাস থেকে ইনহেরিট করা সাব-ক্লাসে annotation ঘটায়। এজন্য যখন সাব-ক্লাস-এর জন্য নির্দিষ্ট কোন annotation এর রিকোয়েস্ট আসে এবং ঐ annotation সাব-ক্লাসে উপস্থিত না থাকে তবে সুপার ক্লাসকে চেক করা হয়। যদি ঐ সুপার ক্লাসে annotation উপস্থিত থাকে এবং ইহা @ Inherited দ্বারা অ্যানোটেটেড (Annotated) থাকে তবে ঐ annotation কে রিটার্ন করা হয়।

@ Override ঃ @ Override এক ধরনের মার্কার অ্যানোটেশন যা শুধুমাত্র মেথড এ ব্যবহৃত হয়। @ Override দ্বারা অ্যানোটেটেড (Annotated) কোন মেথড অবশ্যই সুপার ক্লাসের কোন মেথড override করবে। যদি এটা না হয় তবে ফলাফলে একটি কম্পাইল টাইম এরর আসবে। এজন্য সাধারণত এটা নিশ্চিত করা হয় যে, সুপার ক্লাস মেথডটি যেন সাধারণভাবে ওভারলোডেড (Overloaded) না থেকে প্রকৃতপক্ষে ওভাররিডেন অবস্থায় থাকে।

@ Deprecated ঃ @ Deprecated একটি মার্কার (Marker) অ্যানোটেশন। এটি কোন ডিক্লারেশন obsolete আছে কি না এবং নতুন ফর্মে রিপ্লেস হয়েছে কি না তা নির্দেশ করে।

@ SuppressWarnings ঃ কম্পাইলার দ্বারা ইস্যুকৃত এক বা একাধিক warnings, suppressed হয়েছে কি না এটা নির্ধারণ করে SuppressWarnings। যে warning টি suppress হবে সেটার নাম স্ট্রিং ফর্মে স্পেসিফাই করা হয়। এই অ্যানোটেশনটি যে কোন টাইপ ডিক্লারেশন এর ক্ষেত্রে ব্যবহৃত হয়।

১১.৭ অ্যানোটেশন ডিক্লারেশনে সতর্কতাসমূহ (Restrictions of Annotations) ঃ

অ্যানোটেশন ডিক্লারেশন এর ক্ষেত্রে নিম্নবর্ণিত সতর্কতাসমূহ মেনে চলতে হবে। যথা ঃ

- কোন অ্যানোটেশন অন্য অ্যানোটেশনকে ইনহেরিট (Inherit) করতে পারবে না।
- অ্যানোটেশনের মাধ্যমে ডিক্লেয়ার করা সব মেথড অবশ্যই প্যারামিটার (parameter) বিহীন হবে।
- অ্যানোটেশন generic হতে পারবে না।
- অ্যানোটেশন মেথড কোন 'throws' ক্লাসকে স্পেসিফাই করতে পারে না।
- অ্যানেটেশনের রিটার্ন টাইপ হিসেবে প্রিমিটিভ টাইপ (int, doubble), স্ট্রিং অথবা ক্লাস টাইপের কোন অবজেক্ট, enum টাইপ, অন্য কোন অ্যানোটেশন টাইপ অথবা উল্লিখিত যে কোন টাইপের অ্যারে ইত্যাদির মধ্যে যে কোনটি ব্যবহৃত হবে।

অনুশীলনী-১১

অতি সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। **এনিউমিরেশন (Enumeration) কী?**

উত্তরঃ এনিউমিরেশন (Enumeration) হচ্ছে কতগুলো কন্সট্যান্ট নামের সমষ্টি। মূলত ইউজার ডিফাইন্ড ডাটা টাইপ তৈরির একটি কৌশল হচ্ছে এনিউমিরেশন।

২। **এনিউমিরেশন তৈরির ফরম্যাট লেখ।**

উত্তরঃ এনিউমিরেশন তৈরির ফরম্যাট হল–

```
enum days
}
Saturday, Sunday, Monday, Tuesday, Friday
}
```

এখানে 'enum' হল এনিউমিরেশন তৈরির কী ওয়ার্ড। Saturday, Sunday, Monday, Tuesday, Friday ইত্যাদির প্রতিটিই হল এক একটি কন্সট্যান্ট। প্রতিটি কন্সট্যান্ট 'days' ক্লাসের স্ট্যাটিক ফাইনাল (Static final) মেম্বার এবং পাবলিক হিসেবে ডিক্লেয়ার করা হয়েছে।

৩। **values ( ) এবং values ( ) মেথডের syntax লেখ।**

উত্তরঃ **ভ্যালু মেথড (Value of method)** ঃ ভ্যালু মেথডটি একটি অ্যারে রির্টান করে যা কতগুলো এনিউমিরেশন কনস্ট্যান্ট এর সমষ্টি। এর ডাটা টাইপ হচ্ছে 'enum'। value( ) মেথডের ফরম্যাট নিম্নরূপ–

```
public static enum-type [ ] values ( )
```

ভ্যালু অব মেথড (Value of method) ঃ value of ( ) মেথড একটি স্ট্রিং এনিউমিরেশন কনস্ট্যান্ট এর মান রির্টান করে যা 'str' হিসেবে ব্যবহৃত। value of ( ) মেথডের ফরম্যাট নিম্নরূপ–

```
public static enum-type [ ] value of (string str)
```

৪। **values ( ) এবং value of ( ) method দুটির পার্থক্য লেখ।**

উত্তরঃ values ( ) method টি একটি array return করে, যা কতকগুলো enumeration constant এর সমষ্টি। value of ( ) method টি একটি String enumeration constant এর মান return করে যা str হিসাবে ব্যবহৃত। উভয় ক্ষেত্রেই enumeration এর type হচ্ছে enum-type.

৫। **Type wrappers কী?**

উত্তরঃ Type wrappers হচ্ছে এমন কিছু Data type যেগুলো Object এবং Primitive type এর মধ্যে সমন্বয় সাধন করে। যেমন– double, float, long, integer, short, byte, characher এবং boolean.

৬। **Autoboxing এবং Unboxing কাকে বলে?**

উত্তরঃ যখন কোন Wrapper type এর সমতুল্য একটি Object এর প্রয়োজন হয় তখন যে প্রক্রিয়ায় Primitives type data নিজে নিজেই Encapsulate হয় তাকে Autoboxing বলে।

যে প্রক্রিয়ায় কোন Boxed object এর মান Automatically extracted অর্থাৎ Unboxed হয় তাকে Auto Unboxing বলে।

৭। **Annotations বা Metadata কাকে বলে?**

উত্তরঃ জাভা প্রোগ্রামে সোর্স ফাইলের সাথে সংযুক্ত ঐচ্ছিক তথ্য বা Supplemental information কে Annotation বলা হয়। একটি Annotation কোন Source code generator এর মাধ্যমেও Process করা যেতে পারে। এ বৈশিষ্ট্যগুলোকে Metadata বলা হয়।

৮। **অ্যানোটেশন তৈরির ফরম্যাট লেখ।**

উত্তরঃ ইন্টারফেস (Interface) তৈরিতে অ্যানোটেশনস ডিক্লেয়ার করা যায়। এর ফরম্যাট নিম্নরূপ ঃ

```
@ interface MyAnno
{
String str ( );
int val ( );
}
```

৯। **জাভায় ব্যবহৃত কয়েকটি বিল্ট-ইন অ্যানোটেশনস এর নাম লেখ।**

উত্তরঃ জাভায় ব্যবহৃত কয়েকটি বিল্ট-ইন অ্যানোটেশনস এর নাম হল ঃ @Relation, @Documented, @Target, @Inherited, @Override, @Deprecated ও @SupressWarnings ইত্যাদি।

সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। **values ( ) ও value of ( ) ব্যবহারের মধ্যে পার্থক্য লেখ।**

উত্তর সংকেত ঃ ১১.২ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **উদাহরণসহ Auto-boxing ও Auto-unboxing বর্ণনা কর।**

উত্তর সংকেত ঃ ১১.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **জাভা প্রোগ্রামে Type wrapper ব্যাখ্যা কর ।**

উত্তর সংকেত ঃ ১১.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। **উদাহরণসহ জাভা প্রোগ্রামে ব্যবহৃত বিল্ট-ইন Annotation ব্যাখ্যা কর।**

উত্তর সংকেত ঃ ১১.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। **Annotation ব্যবহারের ক্ষেত্রে সতর্কতাগুলো লেখ।**

উত্তর সংকেত ঃ ১১.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

রচনামূলক প্রশ্নোত্তর ঃ

১। **জাভা প্রোগ্রামে ব্যবহৃত টাইপ র‍্যাপারগুলো বর্ণনা কর।**

উত্তর সংকেত ঃ ১১.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **Buill-in-annotation গুলো কী কী? বর্ণনা কর।**

উত্তর সংকেত ঃ ১১.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **Enumeration ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।**

উত্তর সংকেত ঃ ১১.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। **Auboxing/Unboxing ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।**

উত্তর সংকেত ঃ ১১.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। **Annotations ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।**

উত্তর সংকেত ঃ ১১.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

ইনপুট/আউটপুট অপারেশন প্রক্রিয়া
I/O Operations

অধ্যায় ১২

১২.১ বাইট স্ট্রিম ক্লাস এবং ক্যারেক্টার স্ট্রিম ক্লাস-এর বর্ণনা (Describe the byte stream class and character stream class) ঃ

বাইট স্ট্রিম এবং ক্যারাক্টার স্ট্রিম (Byte stream and character stream) ঃ ইনপুট-আউটপুট বাইট পরিচালনার জন্য বাইট স্ট্রিম ব্যবহার করা সুবিধাজনক। সাধারণত বাইনারি ডাটা রিডিং ও রাইটিং অপারেশন-এর সময় বাইট স্ট্রিম ব্যবহার করা হয়। ইনপুট-আউটপুট ক্যারাক্টার ডাটা অপারেশন-এর জন্য ক্যারাক্টার স্ট্রিম ব্যবহার করা সুবিধাজনক। ইহা ইউনিকোড ব্যবহার করে বিধায় আন্তর্জাতিকভাবে সর্বত্র ব্যবহারযোগ্য। অনেক ক্ষেত্রে বাইট স্ট্রিম-এর চেয়ে ক্যারাক্টার স্ট্রিম বেশি কার্যকরী। জাভা-এর অরিজিনাল ভার্সন (জাভা ১.০)-এ ক্যারাক্টার স্ট্রিম সংযুক্ত ছিল না, ফলে এর আই/ও ছিল বাইট ওরিয়েন্টেড। জাভা ১.১ ভার্সনে ক্যারাক্টার স্ট্রিম সংযুক্ত হয়, যা বাইট ওরিয়েন্টেডের অনেক ক্লাস বা মেথড-এর সাথে সঙ্গতিপূর্ণ নয়। এ কারণে সুবিধাজনক ক্ষেত্রে ব্যবহারের জন্য ক্যারাক্টার স্ট্রিম-এর পুরাতন অব্যবহৃত কোডগুলোকে আপডেট করা হয়। অন্যকথায়, নিম্নস্তরের সমস্ত আই/ও এখনও বাইট ওরিয়েন্টেড। ক্যারাক্টার-বেসড স্ট্রিম সাধারণত কারেক্টারের সুবিধাজনক ও কার্যকরী ব্যবহারকে নিশ্চিত করে।

বাইট স্ট্রিম ক্লাস (Byte Stream Classes) ঃ বাইট স্ট্রিমকে দুই ধরনের ক্লাস হায়ারারকিক্যাল এর মাধ্যমে সজ্ঞায়িত করা যায়। সর্বোচ্চ স্তরে দুটি অ্যাবস্ট্রাক্ট ক্লাস রয়েছে– ইনপুট স্ট্রিম (Input Stream) এবং আউটপুট স্ট্রিম (Output Stream)। প্রত্যেকটি অ্যাবস্ট্রাক্ট ক্লাস-এর আবার কয়েকটি সুনির্দিষ্ট সাব-ক্লাস রয়েছে, যা বিভিন্ন ডিভাইস (যেমন– ডিস্ক ফাইল, নেটওয়ার্ক কানেকশন এমনকি মেমরি বাফার) এর মধ্যে পার্থক্য নিয়ন্ত্রণ করে। টেবিল ১২.১-এ বাইট স্ট্রিম ক্লাস দেখানো হয়েছে।

অ্যাবস্ট্রাক্ট ক্লাসসমূহ Input Stream এবং Output Stream কয়েক ধরনের কী মেথড (Key methode)-কে সংজ্ঞায়িত করে, যা অন্য কোন স্ট্রিম ক্লাসগুলো ব্যবহার করে। এর মধ্যে গুরুত্বপূর্ণ দুটি হচ্ছে Read ( ) এবং Write ( ), যেগুলো ডাটা বাইটকে রিড এবং রাইট করার জন্য ব্যবহৃত হয়। উভয় মেথডই Input Stream এবং Output Stream এর ভিতর অ্যাবস্ট্রাক্ট হিসাবে ঘোষণা করা হয়।

ক্যারেক্টার স্ট্রিম ক্লাস (Character Stream Classes) ঃ ক্যারাক্টার স্ট্রিমকে দুই ধরনের ক্লাস হায়ারারকিক্যাল এর মাধ্যমে সংজ্ঞায়িত করা যায়। সর্বোচ্চ স্তরে দুটি অ্যাবস্ট্রাক্ট ক্লাস রয়েছে ঃ Reader এবং Writer. এই অ্যাবস্ট্রাক্ট ক্লাসগুলো ক্যারাক্টার স্ট্রিম-এর ইউনিকোডকে নিয়ন্ত্রণ করে। এদের প্রত্যেকটির জন্য জাভা-এর কয়েক ধরনের সুনির্দিষ্ট সাব-ক্লাস রয়েছে। ১২.২ নং টেবিলে ক্যারাক্টার স্ট্রিম ক্লাসগুলো দেখানো হয়েছে।

অ্যাবস্ট্রাক্ট ক্লাস Reader এবং Writer কয়েক ধরনের কী মেথড (Key methode) কে সংজ্ঞায়িত করে, যা অন্য স্ট্রিম ক্লাসগুলো ব্যবহার করে। এর মধ্যে দুটি গুরুত্বপূর্ণ মেথড হচ্ছে read ( ) এবং write ( ), যেগুলো যথাক্রমে ক্যারাক্টার ডাটাকে রিড এবং রাইট করার জন্য ব্যবহৃত হয়।

Stream Class	Meaning
BufferedInputStream	Buffered Input Stream
BufferedOutputStream	Buffered Output Stream
ByteArrayInputStream	Input Stream that reads from a byte array
ByteArrayOutputStream	Output Stream that writes to a byte array
DataInputStream	An Input Stream that contains method for reading the java standard data types
DataOutputStream	An Output Stream that contains method for writing the java standard data types
FileInputStream	Input Stream that reads from a file
FileOutputStream	Output Stream that writes to a file
FilterInputStream	Implements Input Stream
FilterOutputStream	Implements Output Stream
InputStream	Abstract Class that describes stream input
ObjectInputStream	Input stream for objects
ObjectOutputStream	Output stream for objects
OutputStream	Abstract Class that describes stream output
PipedInputStream	Input pipe
PipedOutputStream	Output pipe
PrintStream	Output stream that contains print( ) and printIn( )
PushloackInputStream	Input stream that supports one - byte "unget", which returns a byte to the input stream
RandomAccessFile	Supports random access file I/O
SequenceInputStream	Input stream that is a combination of two or more input streams that will be read sequentially, one after the other

টেবিল ১২.১ : The Byte Stream Classes

Stream Class	Meaning
BufferedReader	Buffered Input character Stream
BufferedWrites	Buffered Output character Stream
CharArrayReader	Input Stream that reads from a character array
CharArrayWriter	Output Stream that writes from a character array
FileReader	Input stream that reads from a file
FileWriter	Output stream that reads from a file
FilterReader	Filter reader
FilterWriter	Filter writer
InputStreamReader	Input stream that translates bytes to characters
LineNumberReader	Input stream that counts lines
OutputStreamWriter	Output stream that translates characters to bytes

PipedReader	Input pipe
PipeWriter	Output pipe
PrintWriter	Output stream that contains print( ) and printIn( )
PushbackReader	Input stream that allows characters to be returned to the input stream.
Reader	Abstract class that describes character stream input
StringReader	Input Stream that reads from a string
StringWriter	Output stream that writes to a string
Writer	Abstract class that describes character stream output

টেবিল ১২.২ : The Character Stream I/O Classes

১২.২ রিডিং কনসোল ইনপুট এবং রাইটিং কনসোল আউটপুট (Describe the reading console input and writing console imput) ঃ

রিডিং কনসোল ইনপুট (Reading Console Input) ঃ জাভা ১.০-তে কনসোল (Console) ইনপুট-এ কাজ করার জন্য শুধুমাত্র বাইট স্ট্রিম এবং পুরাতন কোড ব্যবহার করা হতো। বর্তমানেও কনসোল ইনপুট রিড করার জন্য বাইট স্ট্রিম ব্যবহার করা সম্ভব কিন্তু এটি অনুমোদিত নয়। রিডিং কনসোল ইনপুট-এর জন্য গ্রহণযোগ্য মেথড হচ্ছে ক্যারাক্টার ওরিয়েন্টেড স্ট্রিম, যা প্রোগ্রাম মেইনটেইন করাকে অনেক সহজ করে দেয়। জাভাতে System.in হতে রিডিং এর মাধ্যমে কনসোল ইনপুট তৈরি করা হয়। কনসোল-এর সাথে সংযুক্ত ক্যারেক্টার বেসড স্ট্রিম পাওয়ার জন্য BufferedReader অবজেক্ট-এর মধ্যে System.in কে wrap করে। BufferedReader বাফারড ইনপুট সিস্টেমকে সাপোর্ট করে। এটি সাধারণত নিম্নের কন্সট্রাক্টর ব্যবহার করে ঃ

BufferedReader (Reader InputReader)

এখানে ইনপুট রিডার হচ্ছে BufferedReader এর সাথে সংযুক্ত স্ট্রিম। Reader একটি অ্যাবস্ট্রাক্ট ক্লাস। এর একটি বাস্তব সাব-ক্লাস হচ্ছে InputStreamReader, যা বাইটকে ক্যারেক্টার-এ রূপান্তরিত করে।

System.in এর সাথে সংযুক্ত একটি Inputstreamreader পেতে হলে নিম্নলিখিত কনস্ট্রাক্টর ব্যবহার করতে হবে ঃ

InputStreamReader (InputStream inputStream)

কারণ System.in সবসময় InputStream type এর অবজেক্টকে রেফার করে, যা InputStream এর ক্ষেত্রে ব্যবহৃত হয়। নিম্নলিখিত কোডগুলোর সমন্বয়ে কী-ওয়ার্ড-এর সাথে সম্পর্কিত একটি BufferReader তৈরি করা যায় ঃ

BufferReader br = new BufferReader (new InputStreamReader(System.in));

এই স্টেটমেন্ট এক্সিকিউশন হলে কনসোল এর সাথে সংযুক্ত ক্যারেক্টার বেসড স্ট্রিম br System.in এর মাধ্যমে পরিচালিত হবে।

রাইটিং কনসোল আউটপুট (Writing Console Output) ঃ কনসোল আউটপুট সাধারণত পূর্বে বর্ণনাকৃত Print ( ) এবং Println ( ) এর সাথে সম্পর্কিত। এই মেথডগুলোকে PrintStream ক্লাস-এর মাধ্যমে সংজ্ঞায়িত করা হয়, যা System.out অবজেক্টকে রেফার করে। যদিও System.out একটি বাইট স্ট্রিম তবুও এর মাধ্যমে সহজ প্রোগ্রাম আউটপুট তৈরি করা যায়। যদিও PrintStream, OutputStream থেকে প্রাপ্ত আউটপুট স্ট্রিম তথাপি এটি লো-লেভেল মেথড Write ( ) এর ক্ষেত্রে ব্যবহৃত হয়। এভাবে কনসোল রাইট করার জন্য Write ( ) ব্যবহার করা যায়। PrintStream এর মাধ্যমে ডিফাইনকৃত Write ( ) এর সহজ ফর্ম নিম্নে দেয়া হল ঃ

Void Write (int byteval)

Stream Class	Meaning
BufferedInputStream	Buffered Input Stream
BufferedOutputStream	Buffered Output Stream
ByteArrayInputStream	Input Stream that reads from a byte array
ByteArrayOutputStream	Output Stream that writes to a byte array
DataInputStream	An Input Stream that contains method for reading the java standard data types
DataOutputStream	An Output Stream that contains method for writing the java standard data types
FileInputStream	Input Stream that reads from a file
FileOutputStream	Output Stream that writes to a file
FilterInputStream	Implements Input Stream
FilterOutputStream	Implements Output Stream
InputStream	Abstract Class that describes stream input
ObjectInputStream	Input stream for objects
ObjectOutputStream	Output stream for objects
OutputStream	Abstract Class that describes stream output
PipedInputStream	Input pipe
PipedOutputStream	Output pipe
PrintStream	Output stream that contains print( ) and printIn( )
PushloackInputStream	Input stream that supports one – byte "unget", which returns a byte to the input stream
RandomAccessFile	Supports random access file I/O
SequenceInputStream	Input stream that is a combination of two or more input streams that will be read sequentially, one after the other

টেবিল ১২.১ : The Byte Stream Classes

Stream Class	Meaning
BufferedReader	Buffered Input character Stream
BufferedWrites	Buffered Output character Stream
CharArrayReader	Input Stream that reads from a character array
CharArrayWriter	Output Stream that writes from a character array
FileReader	Input stream that reads from a file
FileWriter	Output stream that reads from a file
FilterReader	Filter reader
FilterWriter	Filter writer
InputStreamReader	Input stream that translates bytes to characters
LineNumberReader	Input stream that counts lines
OutputStreamWriter	Output stream that translates characters to bytes

এই মেথড-এর মাধ্যমে স্ট্রিমে বাইট লিখার ক্ষেত্রে বাইটকে byteval দ্বারা স্পেসিফাই করা হয়। যদিও byteval-কে ইন্টিজার হিসাবে ডিক্লেয়ার হয়, কিন্তু লিখার সময় শুধুমাত্র এর লো-অর্ডার আট বিট ব্যবহার করা হয়।

নিম্নে Write ( ) ব্যবহার করে ক্যারেক্টার "A" এর আউটপুট নির্ণয়ের একটি উদাহরণ দেখানো হলঃ

```
// Demonstrate System.out.write ( )
    Class WriteDemo {
    Public Static Void Main (string args[]) {
    int b;
    b = 'A';
    System.out.write (b);
    Syetem.out.write ('int')
    }
}
```

তবে কনসোল আউটপুট-এর ক্ষেত্রে Write ( ) সবসময় ব্যবহার করা যায় না; কারণ Print ( ) এবং Println ( ) , Write ( ) এর চেয়ে সহজতর বিধায় কনসোল আউটপুট-এ Print ( ) এবং Println ( )-ই বেশি ব্যবহৃত হয়।

১২.৩ কন্সট্রাক্টর মেথড ব্যবহার করে ফাইল অবজেক্ট তৈরি করা (Mention the constructors for creating file objects) ঃ

জাভা ভাষায় ডাটা ফাইল খোলার জন্য প্রথমে একটি ফাইল স্ট্রিম অবজেক্ট তৈরি করা হয়। অতঃপর তা সংশ্লিষ্ট ফাইলের সাথে লিংক করতে হয়। ফাইল স্ট্রিম অবজেক্ট তৈরি করতে File Writer, Print writer File Reader প্রভৃতি ক্লাস এবং তাদের কনস্ট্রাক্টর মেথড ব্যবহৃত হয়। এগুলোর মধ্যে কখন কোনটি ব্যবহৃত হবে তা নির্ভর করে ফাইল মোড এবং ফাইল ব্যবহারের উদ্দেশ্যের উপর। অর্থাৎ নতুন ফাইল তৈরি করা হবে নাকি পুরাতন (পূর্বে তৈরি করা) ফাইল খোলা হবে কিংবা ব্যবহৃত ফাইলে ডাটা লেখা (ইনপুট করা) হবে নাকি ফাইল হতে প্রোগ্রামে ডাটা পড়া (আউটপুট করা) হবে প্রভৃতি বিষয়ের উপর। যেমন– ইনপুট ফাইলের (ফাইলের লেখার জন্য) স্ট্রিম অবজেক্ট হিসাবে সাধারণত File writer ক্লাস ও তার কনস্ট্রাক্টর মেথড File writer ( ) এবং ইনপুট ফাইলের (ফাইল থেকে পড়ার জন্য) স্ট্রিম অবজেক্ট হিসেবে সাধারণত File Reader ক্লাস ও তার কনস্ট্রাক্টর মেথড File Reader ( ) ব্যবহৃত হয়।

কনস্ট্রাক্টর মেথড ব্যবহার করে নতুন কিংবা পুরাতন ডাটা ফাইল খোলার ফরম্যাট হল ঃ

```
File stream class file stream object = new file stream class )=("File path", "File Name");
```

যেমন–

```
File writer fw = new File writer ("C// test.doc");
```

উল্লেখ্য, যে কোন ডাটা ফাইলের কাজ শেষে তা বন্ধ করতে হয়। অন্যথায় তা পুনরায় খুলতে সমস্যা হয়। ডাটা ফাইল বন্ধ করলে তা ফাইল স্ট্রিম থেকে আলাদা হয়ে যায়। অতঃপর তা একই কিংবা অন্য কোন ফাইন স্ট্রিমের সাথে সংযুক্ত করা যায়। নিম্নে কনস্ট্রাক্টর থেড ব্যবহার করে ডাটা ফাইল তৈরির জন্য একটি প্রোগ্রাম দেওয়া হল ঃ

```
/* Opening a New Data File*/
import java. io.*;
Public Class File
{
Public static void main (string [ ] args)
Throws IO Exception
{
File writer fr = new file writer ("C : \\ test .doc");
Print wrier Pr = new print writer (fr);
fr. close ( );
```

উপরোক্ত প্রোগ্রামটি নির্বাহ করলে c ড্রাইভে test নামে একটি ডকুমেন্ট (.doc) ফাইল তৈরি হবে।

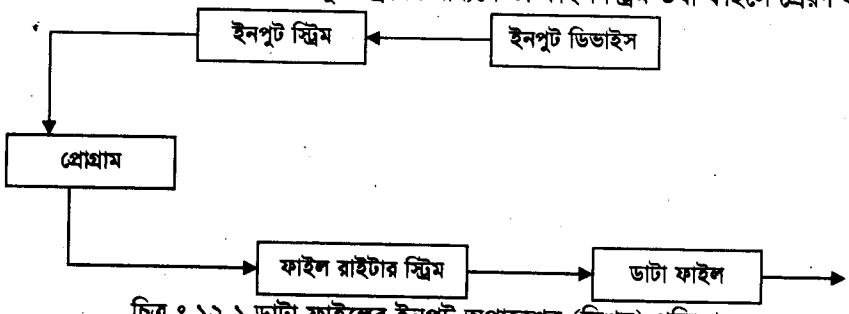
উল্লেখ্য, File writer ক্লাসের File writer ( ) কনস্ট্রাক্টর মেথডটি ব্যবহার কালে ইক্সেপশন প্রদান করতে পারে বিধায় main ( ) মেথডের শেষে অতিরিক্ত Throws IO Exception স্টেটমেন্ট ব্যবহার করা হয়েছে। অবশ্য Throws IO Exception স্টেটমেন্টের পরিবর্তে try ও catch ব্লক ব্যবহার করা যেতে পারে। নিম্নে এরূপ একটি প্রোগ্রাম দেয়া হল ঃ

```
/* Creating a New Data file*/
improt jave . io.*;
Public class File
{
Public static void main (string [ ] args)
{
try {
File writer fr = new file writer ("C : \\jp\ test. txt");
Print writer Pr = new print writer (fr);
fr. close ( );
}
Catch (IO Exception. e)
{
System. Out. Print (" Error :" +e)
System. exit (1); // for normal termination
}
}
}
```

১২.৪ জাভা ফাইলে ইনপুট/আউটপুট অপারেশন (Describc Reading/writing files in Java) ঃ

জাভা প্রোগ্রামে ব্যবহৃত ডাটা ফাইলসমূহের ইনপুট/আউটপুট অপারেশনের জন্য বিশেষ কতগুলো ইনপুট/আউটপুট ও ফাইল স্ট্রিম ক্লাস এবং তাদের বিভিন্ন কনস্ট্রাক্টর ও মেম্বার ফাংশন যুগপৎভাবে ব্যবহৃত হয়। নিম্নে ডাটা ফাইলের ইনপুট ও আউটপুট প্রক্রিয়া বর্ণনা করা হল ।

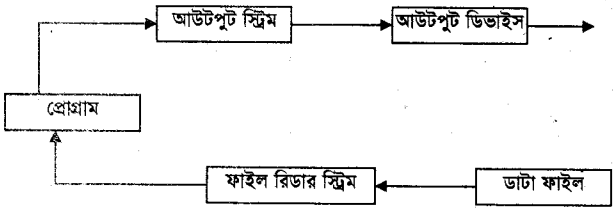
ইনপুট অপারেশন (Input Operation) ঃ ডাটা ফাইলে ইনপুট তথা কিছু লেখার জন্য প্রথমে Writer ক্লাসের উপযুক্ত কোন সাব-ক্লাস (যেমন– FileWriter, PrintWriter, CharArrayWriter, PipeWriter প্রভৃতির যে কোন একটি) ব্যবহার করে নতুন ডাটা ফাইল তৈরি করতে হয় কিংবা পুরাতন ডাটা ফাইল খুলতে হয়। অতঃপর OutputStream ক্লাসের উপযুক্ত কোন সাব-ক্লাস (যেমন– FileOutputStream) ও তার কনস্ট্রাক্টর মেথড ব্যবহার করে সংশ্লিষ্ট ফাইলকে আউটপুট স্ট্রিমের সাথে লিংক করতে হয়। পরিশেষে .. ক্লাসের উপযুক্ত কোন সাব-ক্লাস (যেমন– BufferedReader, InputStreamReader, FileReader প্রভৃতির যে কোন একটি) ব্যবহার করে কী-বোর্ড কিংবা অন্য কোন মাধ্যম হতে ডাটা গ্রহণ করে আউটপুট স্ট্রিমের মাধ্যমে তা ফাইল স্ট্রিম তথা ফাইলে প্রেরণ করা হয়।



চিত্র ঃ ১২.১ ডাটা ফাইলের ইনপুট অপারেশন (লিখন) প্রক্রিয়া

আউটপুট অপারেশন (Output Operation) ঃ ডাটা ফাইলে ইনপুট তথা কিছু লেখার জন্য প্রথমে Reader ক্লাসের উপযুক্ত কোন সাব-ক্লাস (যেমন– FileReader, CharArrayReader, StringReader, PipeReader প্রভৃতির যে কেন একটি) ব্যবহার করে পূর্বে তৈরি করা ডাটা ফাইল খুলতে হয়। তবে তাদের কোনটি কখন ব্যবহৃত হবে তা ফাইলের ধরন এবং ব্যবহারের প্রকৃতির উপর নির্ভর করে। অতঃপর . ক্লাসের উপযুক্ত কোন সাব-ক্লাস (যেমন– BufferedReader, InputStreamReader, FileReader প্রভৃতির যে কোন একটি) ও তার কনস্ট্রাক্টর মেথড ব্যবহার করে সংশ্লিষ্ট ফাইলকে ইনপুট স্ট্রিমের সাথে লিংক করতে হয়। পরিশেষে read ( ) বা readLine ( ) মেথড ব্যবহার করে ফাইল হতে ডাটা গ্রহণ করা হয় এবং পরিশেষে print ( ) বা অন্য কোন উপযুক্ত মেথড ব্যবহার করে সেগুলো মনিটরে প্রদর্শন করা হয় কিংবা উপযুক্ত মেথড ব্যবহার করে প্রিন্টারে বা অন্য কোন ডিভাইসে পাঠান হয়।

নিম্নে আউটপুট অপারেশন (পঠন) প্রক্রিয়া দেখান হল ঃ



চিত্র ঃ ১২.২ ডাটা ফাইলের আউটপুট অপারেশন (পঠন) প্রক্রিয়া

১২.৫ একটি পূর্ণ জাভা স্ট্রিম এর ফ্লো-চার্টসহ বর্ণনা (Describe flowchart of a complete java stream) ঃ

যে কোন আই/ও অপারেশন এর জন্য প্রোগ্রাম এবং কম্পিউটারের আই/ও ডিভাইসের (যেমন– কী-বোর্ড, মনিটর, ডিস্ক ড্রাইভ, টেপ ড্রাইভ, প্রিন্টার, টার্মিনাল ইত্যাদি) মধ্যে নিবিড় সংযোগ থাকা আবশ্যক। প্রোগ্রাম এসব আই/ও ডিভাইসের মধ্যে সংযোগ স্থাপনের মূলে রয়েছে এক বিশেষ ধরনের স্ট্রিম ক্লাস ও তাদের বিভিন্ন মেথড ও অবজেক্ট।

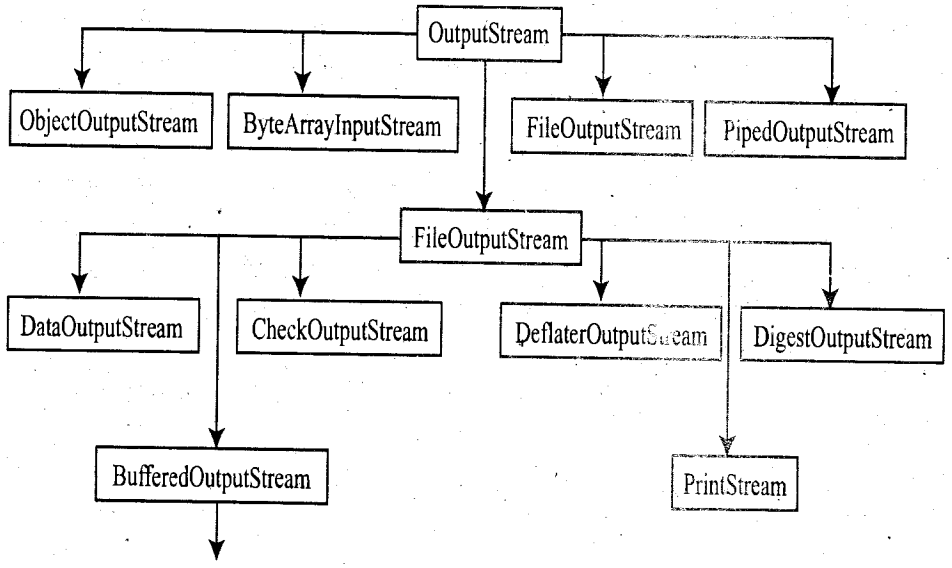
স্ট্রিমসমূহ হচ্ছে একগুচ্ছ বাইটের সমষ্টি। স্ট্রিম দুই ধরনের হতে পারেঃ

- আই/ও স্ট্রিম ক্লাস (I/O Stream Class) এবং
- ফাইল স্ট্রিম ক্লাস (File Stream Class)

আই/ও স্ট্রিম হলো ইনপুট/আউটপুট সম্পাদনকারী একগুচ্ছ বাইটের সমষ্টি; আর ক্লাস স্ট্রিম হলো পরস্পর সম্পর্কযুক্ত কতগুলো ক্লাসের সমষ্টি।

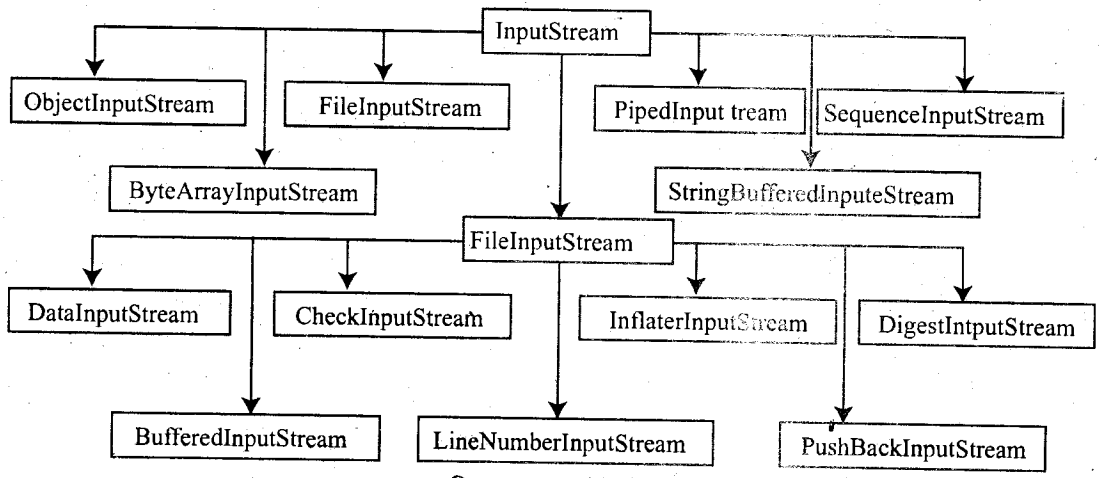
আই/ও স্ট্রিম (I/O Stream Class) ঃ জাভা ভাষায় ইনপুট ও আউটপুট অপারেশনের জন্য প্রধানত InputStream এবং OutputStream এই দুটি স্ট্রিম ক্লাস এবং তাদের সাব-ক্লাসসমূহ ব্যবহৃত হয়। InputStream এবং OutputStream ক্লাস দুটি অ্যাবস্ট্রাক্ট ক্লাস হওয়ায় প্রোগ্রামে সরাসরি এগুলোর কোন অবজেক্ট ঘোষণা করা যায় না। সুতরাং এগুলোর বিভিন্ন সাব-ক্লাসের কনস্ট্রাক্টর ও বিভিন্ন মেথড ব্যবহার করে ইনপুট/আউটপুট অবজেক্ট স্ট্রিম ঘোষণা করা হয়। যেমন– InputStream এবং OutputStream ক্লাসদ্বয়ের সাব-ক্লাস DataInputStream এবং DataOutputStream-এর read( ) এবং write( ) মেথড ব্যবহার করে যথাক্রমে byte টাইপ ভেরিয়েবলের মান গ্রহণ (ইনপুট) এবং প্রদর্শন (আউটপুট) করা যায়। নিম্নে চিত্রে যথাক্রমে InputStream এবং OutputStream ক্লাসদ্বয়ের বহুল ব্যবহৃত সাব-ক্লাসসমূহের ইনহেরিটেন্স রিলেশনশীপ দেখানো হয়েছে। এসব ক্লাসের প্রোটোটাইপ java.io প্যাকেজে বর্ণিত আছে বিধায় প্রোগ্রামে এদের যে কোনটি ব্যবহারকালে প্রোগ্রামের শুরুতে import স্টেটমেন্টের মাধ্যমে এই প্যাকেজে সংযুক্ত করতে হয়।

আউটপুট স্ট্রিম ঃ



চিত্র ঃ ১২.৩ আউটপুট স্ট্রিম

ইনপুট স্ট্রিম ঃ

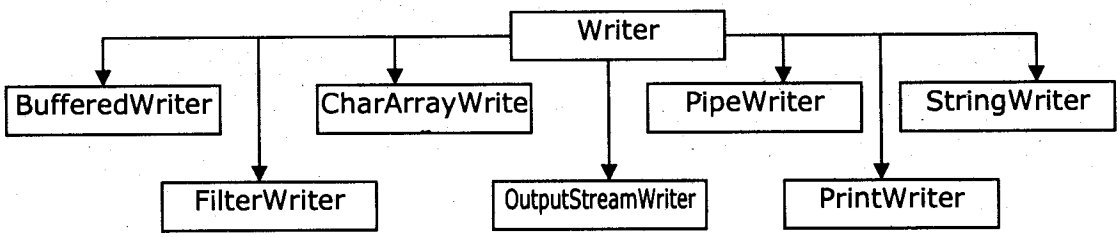


চিত্র ঃ ১২.৪ আউটপুট স্ট্রিম

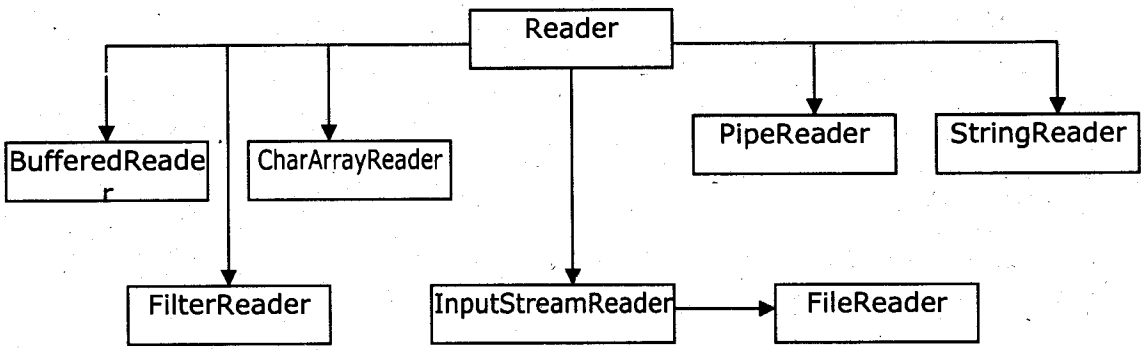
আই/ও স্ট্রিমসমূহ প্রোগ্রাম এবং বিভিন্ন আই/ও ডিভাইসের মধ্যে ডাটা আদান-প্রদানের জন্য ইনটারফেস হিসেবে কাজ করে। সুতরাং স্ট্রিমসমূহকে ডাটা ইনপুট করার জন্য সোর্স স্ট্রিম হিসেবে অথবা ডাটা পাঠানোর জন্য ডেস্টিনেশন স্ট্রিম হিসেবে ব্যবহার করা হয়। প্রোগ্রামে ডাটা সরবরাহ করার জন্য ব্যবহৃত সোর্স স্ট্রিমকে ইনপুট স্ট্রিম এবং প্রোগ্রাম হতে ডাটা পাঠানোর জন্য ব্যবহৃত ডেস্টিনেশন স্ট্রিমকে আউটপুট স্ট্রিম বলা হয়। অর্থাৎ প্রোগ্রামে সোর্স স্ট্রিমের মাধ্যমে কী বোর্ড, মাউস, ফাইল কিংবা অন্য কোন ইনপুট ডিভাইস দ্বারা ডাটা গ্রহণ করা হয় এবং প্রয়োজনীয় প্রক্রিয়াকরণ শেষে মনিটর, প্রিন্টার, ফাইল বা অন্য কোন আউটপুট ডিভাইসের মাধ্যমে আউটপুট স্ট্রিমে ডাটা ফেরত পাঠায়।

ফাইল স্ট্রিম ক্লাস (File Stream Class) ঃ জাভা ভাষায় ডাটা ফাইল সিস্টেমকে বেসিক ইনপুট/আউটপুট অপারেশনের জন্য হায়ারারকিক্যাল ক্লাস ব্যবহৃত হয়, এসব ক্লাসকে ফাইল স্ট্রিম ক্লাস বলা হয়। ফাইল স্ট্রিম ক্লাসসমূহের মধ্যে Writer এবং Reader অন্যতম। Writer এবং Reader ক্লাস দুটি অ্যাবস্ট্রাক্ট ক্লাস হওয়ায় প্রোগ্রামে সরাসরি এগুলোর কোন অবজেক্ট ঘোষণা করা যায় না। সুতরাং এগুলোর বিভিন্ন সাব-ক্লাসের কনস্ট্রাক্টর ও বিভিন্ন মেথড ব্যবহার করে ফাইল ইনপুট/আউটপুট অবজেক্ট স্ট্রিম ঘোষণা করা হয়। যেমন– Writer এবং Reader ক্লাসদ্বয়ের সাব-ক্লাস FileWriter এবং FileReader এর কনস্ট্রাক্টর মেথড FileWriter( ) এবং FileReader( ) ব্যবহার করে যথাক্রমে ডাটা ফাইলে ইনপুট (লেখা) এবং আউটপুট (পড়া) অপারেশনের জন্য যথাক্রমে ইনপুট ও আউটপুট ফাইল স্ট্রিম তৈরি করা হয়। নিম্নে যথাক্রমে Writer এবং Reader ক্লাসদ্বয়ের বহুল ব্যবহৃত সাব-ক্লাসসমূহের ইনহেরিটেন্স রিলেশনশীপ দেখানো হল ।

এসব ক্লাসের প্রটোটাইপ java.io প্যাকেজে বর্ণিত আছে বিধায় প্রোগ্রামে এদের যে কোনটি ব্যবহার কালে প্রোগ্রামের শুরুতে import স্টেটমেন্টের মাধ্যমে এই প্যাকেজে সংযুক্ত করতে হয়।



Writer স্ট্রিম ক্লাসের সাব-ক্লাসসমূহের ইনহেরিটেন্স রিলেশনশীপ



Reader স্ট্রিম ক্লাসের সাব-ক্লাসসমূহের ইনহেরিটেন্স রিলেশনশীপ

১২.৬ র‍্যানডম ফাইল অ্যাক্সেসিং (Random File Accessing) ঃ

কোন ফাইল থেকে ডাটা আইটেম বা অবজেক্টের ইনপুট বা আউটপুট অপারেশনকে ফাইল অ্যাক্সেসিং (File Accessing) বলা হয়। সাধারণত ফাইলের ডাটা আইটেম বা অবজেক্ট যদি একটি ক্যারেক্টার করে পড়া হয় তবে ফাইলের বিশেষ কোন অবস্থানের ডাটা বা অবজেক্ট পড়ার জন্য পূর্ববর্তী ডাটা বা অবজেক্টসমূহ পড়ে আসতে হয় । ফাইলের এরূপ অ্যাক্সেসিং পদ্ধতিকে ধারাবাহিক বা সিকুয়েন্সিয়াল ফাইল অ্যাক্সেসিং (Sequential File Accessing) বলা হয়। সিকুয়েন্সিয়াল পদ্ধতি ধীরগতিসম্পন্ন। তাই ফাইলের যে কোন অবস্থান হতে আরও সহজে ডাটা পড়া বা লেখার জন্য অপর একটি পদ্ধতি ব্যবহৃত হয়, যা র‍্যানডম ফাইল অ্যাক্সেসিং (Random File Accessing) নামে পরিচিত। সাধারণত যেসব ক্ষেত্রে ফাইলে নিম্নের এক বা একাধিক কার্যক্রম চালানোর প্রয়োজন হয় সেসব ক্ষেত্রে র‍্যানডম ফাইল অ্যাক্সেসিং পদ্ধতি ব্যবহৃত হয় ঃ

- ফাইলের ডাটা বা অবজেক্টসমূহ প্রদর্শন করা
- নতুন ডাটা বা অবজেক্ট যোগ করা
- পুরাতন ডাটা বা অবজেক্ট পরিবর্তন বা পরিবর্ধন করা
- পুরাতন ডাটা বা অবজেক্ট মুছে দেয়া, ইত্যাদি।

র‍্যানডম ফাইল অ্যাক্সেসিং পদ্ধতিতে ফাইল পয়েন্টার ফাইলের প্রয়োজনীয় স্থানে স্থানান্তরের মাধ্যমে কাঙ্ক্ষিত অবজেক্ট রিড বা রাইট করা হয়। jav.io প্যাকেজের অন্তর্গত RandomAccessFile নামে এক বিশেষ ধরনের ক্লাস ব্যবহৃত হয়। এটি InputStream কিংবা InputStream ক্লাসের কোন ডিরাইভড ক্লাস নয়, তবে DataInput এবং DataOutput নামে দুটি ইন্টারফেস বাস্তবায়ন করে। র‍্যান্ডম ফাইল অ্যাক্সেসিং-এর জন্য ব্যবহৃত RandomAccessFile ক্লাসের দুটি কনস্ট্রাক্টর মেথড আছে ।

RandomAccessFile (File FileObject, String AccessMode) throws FileNotFoundException
RandomAccessFile (String FileName, String AccessMode) throws FileNotFoundException

এখানে FileObject ও FileName যথাক্রমে File ক্লাসের অবজেক্ট ও ফাইলের নাম নির্দেশ করে এবং AccessMode ফাইলের অ্যাক্সেস মোড নির্দেশ করে। অ্যাক্সেস মোড "wr" হলে রাইট (ইনপুট) ও রিড (আউটপুট) উভয় মোডে ফাইল ওপেন হয়। সুতরাং, এরূপ ফাইলে একইসাথে ডাটা লেখা ও পড়া যায়। আর অ্যাক্সেস মোড "r" হলে কেবল রিড (আউটপুট) মোডে ফাইল ওপেন হয়। সুতরাং, এরূপ ফাইল থেকে কেবল পড়া যায় কিন্তু ফাইলে কিছু লেখা যায় না।

র‍্যানডম ফাইল অ্যাক্সেসিং পদ্ধতি বাস্তবায়নের জন্য RandomAccessFile ক্লাসের seek ( ) নামে একটি মেথড ব্যবহৃত হয়, যা ফাইল পয়েন্টারে নতুন অবস্থান নির্ধারণ করে। seek ( ) মেথডের সাধারণ রূপ হল ঃ

void seek (long NewPosition) throws IOException

এখানে NewPosition বাইটে প্রকাশিত ফাইল পয়েন্টারের নতুন অবস্থান নির্দেশ করে, যা ফাইলের প্রথম হতে গণনা শুরু হয়। প্রতিবার seek ( ) মেথড নির্বাহের পরে পরবর্তী নির্বাহের জন্য ফাইল পয়েন্টার নতুন অবস্থানে স্থানান্তরিত হয়।

১২.৭ ডাটা ফাইলে অবজেক্টের ইনপুট/আউটপুট অপারেশন (The object I/O Operation in Data File) ঃ

ডাটা ফাইলে অবজেক্ট ইনপুট করার বা (লেখার) জন্য গৃহীত অবজেক্টকে অপেক্ষাকৃত ছোট অংশে বিভক্ত করা হয়। এজন্য StringTokenizer ( ) ক্লাসের একটি অবজেকট t ব্যবহৃয়ে হয়েছে, যাকে স্ট্রিং টোকেনাইজার অবজেক্ট (String Tokenizer Object) বলা হয়। স্ট্রিং টোকেনাইজার অবজেক্ট ঘোষণার সময় অবজেক্টের দুটি অংশ বা দুটি স্ট্রিং পৃথকীকরণের জন্য ফাঁকা স্থান বা অন্য বিশেষ ক্যারেক্টার (যেমন, ভার্টিক্যাল বার (|), ড্যাশ (-)), কিংবা একটি ক্যারেক্টারগুচ্ছ (যেমন– "-" "\n\t") ইত্যাদি ব্যবহার করা যায়। এরূপ ক্যারেক্টার বা ক্যারেক্টারগুচ্ছকে ডিলিমিটার ক্যারেক্টার (delimeter character) বলা হয়। ডিলিমিটার হিসেবে ব্যবহৃত ক্যারেক্টার বা ক্যারেক্টারগুচ্ছ দ্বারা ডাটা ফাইলে স্ট্রিং বা অবজেক্টের প্রোগ্রামসমূহ আলাদা হয়। নিম্নে ডাটা ফাইলে অবজেক্টের ইনপুট ও আউটপুট অপারেশন সম্পর্কিত একটি প্রোগ্রাম দেয়া হল ঃ

```
/* Writing and Reading Objects in Data File */
import java.io.*;
import java.util.*;
class Student {
int Roll;
String Name;
```

```
double Mark;
public Student ( int R, String N, double M)
{
Roll=R;
Name=N;
Mark=M;
}
public Student ( )          //Empty Constructor
{
}
public void WriteData ( PrintWriter PR) throws IOException
{
PR.println(Roll+ "|"+Name+ "|"+Mark);        // | as delimeter
}
public void ReadData (BufferedReader in) throws IOException
{
String Str = in.readLine ( );
StringTokenizer t=new StringTokenizer(Str, "|");
Roll = Integer.parseInt (t.nextToken ( ));
Name = t.nextToken ( );
Mark=Double.parseDouble (t.nextToken ( ));
}
public void Display ( )
{
System.out.println( Roll + "  " + Name + "   " +Mark);
}
}
public class ObjectInputOutput
{
static void WriteData(Student [] s, PrintWriter PR) throws IOException
{
PR.println(S.length);
int i;
for ( i= 0; i<S.length; i++)
{
S[i].WriteData(PR);
}
}
static Student [ ] ReadData (BufferedReader BR) throws IOException
{
int n=Integer.parseInt(BR.readLine ( ) );
Student S[] =new Stuent [n];
int i;
for (i=0;i<n;i++)
{
S [ i ]= new Student ( );
S [i ].ReadData (BR);
```

```
}
return S;
}
public static void main (String [] args)
{
Student Std[]=new Student [4];
Std[0]=new Student(101, "Mira" ,70.5);
Std[1]=new Student(102, "Karim" ,73.5);
Std[2]=new Student(103, "Murad" ,50.5);
Std[3]=new Student(104, "Poly" ,60.5);
int i;
try {
PrintWriter PR = new PrintWriter(new FileWriter("c:\\student"));
WriteData(Std,PR);
PR.close( );
}
catch (IOException e)
{
System.out.print ("Error :" + e);
System.exit (1);
}
try{
BufferedReader BR = new BufferedReader (new FileReader ("c:\\student"));
Student s[]=ReadData (BR);
System.out.println("Roll Name  Mark");
System.out.println("````````````````````````````");
for (i=0; i<S.length; i ++ )
{
S.[i].Display( );
}
BR.close ( );
}
catch (IOException e)
{
System.out.println("Error :"  + e);
System.exit (1):
}
}
}
output:
Roll        Name  Mark
````````````````````````
101   Mira   70.5
102   Karim 73.5
103   Murad 50.5
104,   Poly   60.5
```

পলিটেকনিকের সকল বই ডাওনলোড করতে
ভিজিটঃ
www.BDeBooks.Com/polytechnic
```

অনুশীলনী-১২

অতি সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। **Stream কী? এটি কত প্রকার ও কী কী?**

উত্তরঃ জাভা প্রোগ্রামে I/O operation সম্পন্ন করার জন্য ব্যবহৃত একগুচ্ছ বাইট এর সমষ্টিকে Stream বলে। Stream দুই ধরনের হতে পারে, যথা–

I/O stream class এবং

File stream class.

২। **File Accessing কী?**

উত্তরঃ কোন ফাইল থেকে data item বা অবজেক্টের Input বা Output অপারেশনকে ফাইল অ্যাক্সেসিং বলা হয়।

৩। **Sequential File Accessing কী?**

উত্তরঃ সাধারণত ফাইলের ডাটা আইটেম বা অবজেক্ট যদি একটি ক্যারেক্টার করে পড়া হয় তবে ফাইল এর বিশেষ কোন অবস্থানের ডাটা বা অবজেক্ট পড়ার জন্য পূর্ববর্তী ডাটা বা অবজেক্টসমূহ পড়ে আসতে হয়। ফাইলের এরূপ অ্যাক্সেসিং পদ্ধতিকে Sequential বা ধারাবাহিক ফাইল অ্যাক্সেসিং বলে।

৪। **Byte stream ও Character stream বলতে কী বোঝায়?**

উত্তরঃ Byte হিসাবে Input/output provide করার জন্য Byte stream ব্যবহৃত হয়। সাধারণত বাইনারি ডাটা রিডিং ও রাইটিং অপারেশন এর সময় বাইট Stream ব্যবহার করা হয়। ইনপুট-আউটপুট ক্যারেক্টার ডাটা অপারেশন এর জন্য ক্যারেক্টার Stream ব্যবহার করা হয়।

৫। **Java stream এর সুপার ক্লাসগুলো কী কী?**

উত্তরঃ Java তে Stream এর জন্য চারটি সুপার ক্লাস রয়েছে। এগুলো হচ্ছে Inputstream, Outputstream, Reader এবং Writer.

৬। **Random Access File এর Constructor method-গুলো কী কী?**

উত্তরঃ Random Access File Class এর দুটি Construct or Method রয়েছে। যথা– Randon Access File in = New Random Access File ("filename.ext ", "r");
Randoma Access file inout = New Access File ("filename. ext", "rw")

৭। **Data file খোলার Format লিখ।**

উত্তরঃ Date file তৈরি বা খোলার Formate নিম্নরূপ ঃ

File Stream Class File Stream object = new file stream Class ("File Path", "File Name");

সংক্ষিপ্ত প্রশ্নাবলি ঃ

১। **Reading Console Input এবং Writing Console Output** বলতে কী বোঝায়? উদাহরণ দাও।

উত্তর সংকেত ঃ ১২.২ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **কনস্ট্রাক্টর ব্যবহার করে Data file তৈরি করার পদ্ধতি বর্ণনা কর।**

উত্তর সংকেত ঃ ১২.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **জাভা প্রোগ্রামে ব্যবহৃত ইনপুট এবং আউটপুট অপারেশন বর্ণনা কর।**

উত্তর সংকেত ঃ ১২.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। **Random File Accessing পদ্ধতি ব্যাখ্যা করা।**

উত্তর সংকেত ঃ ১২.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

রচনামূলক প্রশ্নাবলি ঃ

১। **Data file এর Input/Output operation বর্ণনা কর।**

উত্তর সংকেত ঃ ১২.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **Data file এর Object reading এবং Writing এর একটি জাভা প্রোগ্রাম লেখ।**

উত্তর সংকেত ঃ ১২.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **বাইট স্ট্রিম ও ক্যারেক্টার স্ট্রিম ক্লাসগুলো বর্ণনা কর।**

উত্তর সংকেত ঃ ১২.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। **Constructor ব্যবহার করে file object তৈরির জন্য একটি জাভা প্রোগ্রাম লিখ।**

উত্তর সংকেত ঃ ১২.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। **Writing এবং Reading Object-এর জন্য Data File-এ জাভায় একটি প্রোগ্রাম লিখ।**

উত্তর সংকেত ঃ ১২.৭ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। **একটি জাভা স্ট্রিম এর চিত্রসহ বর্ণনা দাও।**

উত্তর সংকেত ঃ ১২.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

সুপার সাজেশনস্

অতি সংক্ষিপ্ত প্রশ্নাবলি ঃ

১। কম্পিউটার গ্রাফিক্স কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

২। গ্রাফিক্সের লাইব্রেরি ফাংশন কী?
উত্তর সংকেত ঃ অনুশীলনী-১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

৩। Pixel কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

৪। রেজোলুশন (Resolution) বলতে কী বুঝ? [বাকাশিবো-২০১২, ২০১৩R, ২০১৪]
অথবা, রেজুলেশন কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।
অথবা, রেজুলেশন কাকে বলে?

৫। initgraph () ফাংশনের কাজ কী? [বাকাশিবো-২০১২, ২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

৬। restorecrtmode () ফাংশনের কাজ কী?
উত্তর সংকেত ঃ অনুশীলনী-১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৯ নং দ্রষ্টব্য।

৭। একটি গ্রাফিক্স কার্ডের রেজোলুশন 1024 × 768 এর অর্থ কী? [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

৮। sector ( ) ফাংশনের ব্যবহার লেখ।
অথবা, Sector ফাংশনের ফরম্যাটসহ সংজ্ঞা লিখ। [বাকাশিবো-২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৬ নং দ্রষ্টব্য।

৯। cleardevice ( ) ফাংশনের কাজ কী?
উত্তর সংকেত ঃ অনুশীলনী-১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৭ নং দ্রষ্টব্য।

১০। delay ( ) ফাংশন কখন ব্যবহৃত হয়?
উত্তর সংকেত ঃ অনুশীলনী-১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৮ নং দ্রষ্টব্য।

১১। sound ( ) ফাংশনের কাজ কী?
উত্তর সংকেত ঃ অনুশীলনী-১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৯ নং দ্রষ্টব্য।

১২। cprintf ( ) ও cscanf ( ) ফাংশনের কাজ কী?
উত্তর সংকেত ঃ অনুশীলনী-১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২০ নং দ্রষ্টব্য।

১৩। putimage ( ) ফাংশনের ব্যবহার ও ফরম্যাট লেখ।
উত্তর সংকেত ঃ অনুশীলনী-১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২২ নং দ্রষ্টব্য।

১৪। movetext ( ) ফাংশনের কাজ কী?
উত্তর সংকেত ঃ অনুশীলনী-১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২৩ নং দ্রষ্টব্য।

১৫। ROM BIOS প্রোগ্রামিং কী?
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

১৬। ইন্টারাপ্ট কী? [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

১৭। সফটওয়্যার ইন্টারাপ্ট বলতে কী বোঝায়? [বাকাশিবো-২০১৩, ২০১৪]
অথবা, সফটওয়্যার ইন্টারাপ্ট কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

১৮। ROM BIOS সার্ভিস কী?
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

১৯। মাউস প্রোগ্রামিং বলতে কী বোঝায়? [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

২০। মাউস প্রোগ্রামিং এর ক্ষেত্রে বিবেচ্য বিষয়সমূহ কী কী? [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

২১। biosequip ( ) ফাংশনের কাজ কী?
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

২২। মেশিন ল্যাংগুয়েজ কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

২৩। এসেম্বলি ল্যাংগুয়েজ কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

২৪। প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

২৫। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং বলতে কী বোঝায়? [বাকাশিবো-২০১৩, ২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

২৬। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

২৭। কয়েকটি অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজের নাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

২৮। অবজেক্ট বলতে কী বোঝায়? [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৭ নং দ্রষ্টব্য।

২৯। ক্লাস কী? [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৮ নং দ্রষ্টব্য।

৩০। ইনহেরিটেন্স কাকে বলে? [বাকাশিবো-২০১২, ২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৯ নং দ্রষ্টব্য।

৩১। ডাটা অ্যাবস্ট্রাকশন কী? [বাকাশিবো-২০১২]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১০ নং দ্রষ্টব্য।

৩২। এনক্যাপসুলেশন বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

৩৩। পলিমরফিজম কী?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১২ নং দ্রষ্টব্য।

৩৪। ম্যাসেজ পাসিং বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৩ নং দ্রষ্টব্য।

৩৫। ডাইনামিক বাইন্ডিং কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৪ নং দ্রষ্টব্য।

৩৬। সি ++ কে হাইব্রিড প্রোগ্রামিং ল্যাংগুয়েজ বলা হয় কেন?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৫ নং দ্রষ্টব্য।

৩৭। Hyper Card কী?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২০ নং দ্রষ্টব্য।

১৭। সফটওয়্যার ইন্টারাপ্ট বলতে কী বোঝায়? [বাকাশিবো-২০১৩, ২০১৪]
অথবা, সফটওয়্যার ইন্টারাপ্ট কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

১৮। ROM BIOS সার্ভিস কী?
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

১৯। মাউস প্রোগ্রামিং বলতে কী বোঝায়? [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

২০। মাউস প্রোগ্রামিং এর ক্ষেত্রে বিবেচ্য বিষয়সমূহ কী কী? [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

২১। biosequip ( ) ফাংশনের কাজ কী?
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

২২। মেশিন ল্যাংগুয়েজ কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

২৩। এসেম্বলি ল্যাংগুয়েজ কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

২৪। প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

২৫। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং বলতে কী বোঝায়? [বাকাশিবো-২০১৩, ২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

২৬। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

২৭। কয়েকটি অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজের নাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

২৮। অবজেক্ট বলতে কী বোঝায়? [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৭ নং দ্রষ্টব্য।

২৯। ক্লাস কী? [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৮ নং দ্রষ্টব্য।

৩০। ইনহেরিটেন্স কাকে বলে? [বাকাশিবো-২০১২, ২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৯ নং দ্রষ্টব্য।

৩১। ডাটা অ্যাবস্ট্রাকশন কী? [বাকাশিবো-২০১২]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১০ নং দ্রষ্টব্য।

৩২। এনক্যাপসুলেশন বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

৩৩। পলিমরফিজম কী?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১২ নং দ্রষ্টব্য।

৩৪। ম্যাসেজ পাসিং বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৩ নং দ্রষ্টব্য।

৩৫। ডাইনামিক বাইন্ডিং কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৪ নং দ্রষ্টব্য।

৩৬। সি ++ কে হাইব্রিড প্রোগ্রামিং ল্যাংগুয়েজ বলা হয় কেন?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৫ নং দ্রষ্টব্য।

৩৭। Hyper Card কী?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২০ নং দ্রষ্টব্য।

১৭। সফটওয়্যার ইন্টারাপ্ট বলতে কী বোঝায়? [বাকাশিবো-২০১৩, ২০১৪]
অথবা, সফটওয়্যার ইন্টারাপ্ট কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

১৮। ROM BIOS সার্ভিস কী?
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

১৯। মাউস প্রোগ্রামিং বলতে কী বোঝায়? [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

২০। মাউস প্রোগ্রামিং এর ক্ষেত্রে বিবেচ্য বিষয়সমূহ কী কী? [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

২১। biosequip ( ) ফাংশনের কাজ কী?
উত্তর সংকেত ঃ অনুশীলনী-২ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

২২। মেশিন ল্যাংগুয়েজ কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

২৩। এসেম্বলি ল্যাংগুয়েজ কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

২৪। প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

২৫। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং বলতে কী বোঝায়? [বাকাশিবো-২০১৩, ২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

২৬। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

২৭। কয়েকটি অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজের নাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

২৮। অবজেক্ট বলতে কী বোঝায়? [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৭ নং দ্রষ্টব্য।

২৯। ক্লাস কী? [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৮ নং দ্রষ্টব্য।

৩০। ইনহেরিটেন্স কাকে বলে? [বাকাশিবো-২০১২, ২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৯ নং দ্রষ্টব্য।

৩১। ডাটা অ্যাবস্ট্রাকশন কী? [বাকাশিবো-২০১২]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১০ নং দ্রষ্টব্য।

৩২। এনক্যাপসুলেশন বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

৩৩। পলিমরফিজম কী?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১২ নং দ্রষ্টব্য।

৩৪। ম্যাসেজ পাসিং বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৩ নং দ্রষ্টব্য।

৩৫। ডাইনামিক বাইন্ডিং কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৪ নং দ্রষ্টব্য।

৩৬। সি ++ কে হাইব্রিড প্রোগ্রামিং ল্যাংগুয়েজ বলা হয় কেন?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৫ নং দ্রষ্টব্য।

৩৭। Hyper Card কী?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২০ নং দ্রষ্টব্য।

৩৮। সি++ কে অবজেক্ট অরিয়েন্টেড ল্যাংগুয়েজ বলা হয় কেন?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

৩৯। কয়েকটি C ++ কম্পাইলারের নাম লিখ।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

৪০। ডাটা স্ট্রিম বলতে কি বুঝ?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

৪১। কনসোল I/O কি? ইহা কত প্রকার ও কী কী? [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

৪২। ফরম্যাটেড I/O অপারেশনের প্রয়োজনীয়তা কী? [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

৪৩। ডট অপারেটর বলতে কী বুঝ?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১০ নং দ্রষ্টব্য।

৪৪। ক্যাসকেডিং আই/ও অপারেটর বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

৪৫। রিকার্সন বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৯ নং দ্রষ্টব্য।

৪৬। রিকার্সিভ ফাংশনের সুবিধা লিখ।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২০ নং দ্রষ্টব্য।

৪৭। ইনলাইন ফাংশন কাকে বলে? [বাকাশিবো-২০১৩, ২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২১ নং দ্রষ্টব্য।

৪৮। ফাংশন আরগুমেন্ট বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২২ নং দ্রষ্টব্য।

৪৯। Constant member function কী?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২৩ নং দ্রষ্টব্য।

৫০। ফাংশন ওভারলোডিং বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২৫ নং দ্রষ্টব্য।

৫১। নেস্টেড মেম্বার ফাংশন বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২৮ নং দ্রষ্টব্য।

৫২। স্ট্যাটিক মেম্বার ফাংশন কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২৯ নং দ্রষ্টব্য।

৫৩। অপারেটর ওভারলোডিং কাকে বলে? [বাকাশিবো-২০১৩R, ২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩০ নং দ্রষ্টব্য।

৫৪। কনস্ট্রাক্টর কাকে বলে? [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩১ নং দ্রষ্টব্য।

৫৫। কপি কনস্ট্রাক্টর বলতে কী বোঝায়? [বাকাশিবো-২০১২, ২০১৩R]
অথবা, কপি কনস্ট্রাক্টরের ফরম্যাটসহ সংজ্ঞা লিখ। [বাকাশিবো-২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩২ নং দ্রষ্টব্য।

৫৬। মাল্টিপল কনস্ট্রাক্টর কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩৩ নং দ্রষ্টব্য।

৫৭। ডেস্ট্রাক্টর বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩৫ নং দ্রষ্টব্য।

৫৮। মাল্টিপল ইনহেরিটেন্স কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩৯ নং দ্রষ্টব্য।

৩৮। সি++ কে অবজেক্ট অরিয়েন্টেড ল্যাংগুয়েজ বলা হয় কেন?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

৩৯। কয়েকটি C ++ কম্পাইলারের নাম লিখ।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

৪০। ডাটা স্ট্রিম বলতে কি বুঝ?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

৪১। কনসোল I/O কি? ইহা কত প্রকার ও কী কী? [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

৪২। ফরম্যাটেড I/O অপারেশনের প্রয়োজনীয়তা কী? [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

৪৩। ডট অপারেটর বলতে কী বুঝ?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১০ নং দ্রষ্টব্য।

৪৪। ক্যাসকেডিং আই/ও অপারেটর বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

৪৫। রিকার্সন বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৯ নং দ্রষ্টব্য।

৪৬। রিকার্সিভ ফাংশনের সুবিধা লিখ।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২০ নং দ্রষ্টব্য।

৪৭। ইনলাইন ফাংশন কাকে বলে? [বাকাশিবো-২০১৩, ২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২১ নং দ্রষ্টব্য।

৪৮। ফাংশন আরগুমেন্ট বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২২ নং দ্রষ্টব্য।

৪৯। Constant member function কী?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২৩ নং দ্রষ্টব্য।

৫০। ফাংশন ওভারলোডিং বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২৫ নং দ্রষ্টব্য।

৫১। নেস্টেড মেম্বার ফাংশন বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২৮ নং দ্রষ্টব্য।

৫২। স্ট্যাটিক মেম্বার ফাংশন কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২৯ নং দ্রষ্টব্য।

৫৩। অপারেটর ওভারলোডিং কাকে বলে? [বাকাশিবো-২০১৩R, ২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩০ নং দ্রষ্টব্য।

৫৪। কনস্ট্রাক্টর কাকে বলে? [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩১ নং দ্রষ্টব্য।

৫৫। কপি কনস্ট্রাক্টর বলতে কী বোঝায়? [বাকাশিবো-২০১২, ২০১৩R]
অথবা, কপি কনস্ট্রাক্টরের ফরম্যাটসহ সংজ্ঞ লিখ। [বাকাশিবো-২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩২ নং দ্রষ্টব্য।

৫৬। মাল্টিপল কনস্ট্রাক্টর কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩৩ নং দ্রষ্টব্য।

৫৭। ডেস্ট্রাক্টর বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩৫ নং দ্রষ্টব্য।

৫৮। মাল্টিপল ইনহেরিটেন্স কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩৯ নং দ্রষ্টব্য।

৫৯। হায়ারারকিক্যাল ইনহেরিটেন্স কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪১ নং দ্রষ্টব্য।

৬০। হাইব্রিড ইনহেরিটেন্স কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪২ নং দ্রষ্টব্য।

৬১। ভার্চুয়াল Base class কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪৩ নং দ্রষ্টব্য।

৬২। অ্যাবস্ট্রাক্ট ক্লাস কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪৪ নং দ্রষ্টব্য।

৬৩। ফাংশন ওভারলোডিংজনিত অনিশ্চিয়তা বা অস্পষ্টতা বলতে কী বোঝায়?
অথাব, কী কী কারণে ফাংশন ওভারলোডিংজনিত অনিশ্চয়তা দেখা দিতে পারে?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪৫ নং দ্রষ্টব্য।

৬৪। Identifier কী? [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৫ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

৬৫। জাভায় ব্যবহৃত 10টি Keyword এর নাম লেখ। [বাকাশিবো-২০১৪]
অথবা, পাঁচটি জাভা কী ওয়ার্ডের নাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৫ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

৬৬। JVM বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৫ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৮ নং দ্রষ্টব্য।

৬৭। GUI মোড কী?
উত্তর সংকেত ঃ অনুশীলনী-৫ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১০ নং দ্রষ্টব্য।

৬৮। লিটারেলস কী? জাভায় কয় ধরনের লিটারেলস ব্যবহৃত হয়? [বাকাশিবো-২০১২]
উত্তর সংকেত ঃ অনুশীলনী-৫ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১২ নং দ্রষ্টব্য।

৬৯। সেপারেটর (\) ব্যবহার করা হয় কেন?
উত্তর সংকেত ঃ অনুশীলনী-৫ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৪ নং দ্রষ্টব্য।

৭০। ডাটা টাইপ বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৬ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

৭১। প্রিমিটিভ ও নন-প্রিমিটিভ ডাটা টাইপ কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৬ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

৭২। জাভায় কয় ধরনের ও কী কী বিল্ট ইন ডাটা টাইপ ব্যবহৃত হয়?
উত্তর সংকেত ঃ অনুশীলনী-৬ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

৭৩। ভেরিয়েবলের ডাইনামিক ইনিশিয়ালাইজেশন কী? [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৬ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

৭৪। স্কোপ ও লাইফ টাইম বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৬ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

৭৫। ইউনারি ও বাইনারি অপারেটর বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৬ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৭ নং দ্রষ্টব্য।

৭৬। এক্সপ্রেশন কী?
উত্তর সংকেত ঃ অনুশীলনী-৬ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৮ নং দ্রষ্টব্য।

৭৭। পাঙ্কচুয়েটর (Punctuator) কী?
উত্তর সংকেত ঃ অনুশীলনী-৬ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৯ নং দ্রষ্টব্য।

৭৮। স্টেটমেন্ট (Statement) কী? [বাকাশিবো-২০১৪]
অথবা, স্টেটমেন্ট বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৬ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১০ নং দ্রষ্টব্য।

৭৯। জাভায় ব্যবহৃত স্পেশাল অপারেটরসমূহের নাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৬ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

৮০। কম্পাউন্ড স্টেটমেন্ট কী?
উত্তর সংকেত ঃ অনুশীলনী-৬ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৬ নং দ্রষ্টব্য।

৮১। শর্টহ্যান্ড অ্যাসাইনমেন্ট অপারেটর কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৬ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৭ নং দ্রষ্টব্য।

৮২। ক্লাস ডিক্লারেশন বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

৮৩। জাভায় ক্লাস ডিক্লারেশনের ফরম্যাট লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

৮৪। জাভায় ক্লাসে অবজেক্ট ডিক্লারেশনের ফরম্যাট রেখ।
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

৮৫। ক্লাস মেম্বার অ্যাক্সেসিবিলিটি মোডগুলোর নাম লেখ। [বাকাশিবো-২০১৩]
অথবা, ক্লাসের অ্যাক্সেসিবিলিটি মোডগুলো লেখ। [বাকাশিবো-২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

৮৬। মেথড কাকে বলে? [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

৮৭। মেথডের উপাদানগুলো উল্লেখ কর। [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৭ নং দ্রষ্টব্য।

৮৮। মেথড ওভারলোডিং কাকে বলে? [বাকাশিবো-২০১৪]
অথবা, মেথড ওভারললোডিং বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৯ নং দ্রষ্টব্য।

৮৯। কনস্ট্রাক্টর কী? [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১০ নং দ্রষ্টব্য।

৯০। ওভারলোডিং কনস্ট্রাক্টর বা কপি কনস্ট্রাক্টর কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

৯১। গারবেজ কালেকশন কী?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১২ নং দ্রষ্টব্য।

৯২। কনস্ট্রাক্টর কত প্রকার ও কী কী?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৩ নং দ্রষ্টব্য।

৯৩। মাল্টিপল কনস্ট্রাক্টর কী?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৪ নং দ্রষ্টব্য।

৯৪। সুপার ক্লাস ও সাব ক্লাস কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

৯৫। সিঙ্গেল ইনহেরিটেন্স কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

৯৬। মাল্টিলেভেল ইনহেরিটেন্স কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

৯৭। হায়ারারকিক্যাল ইনহেরিটেন্স কী?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

৯৮। ওভাররিডেন ও ওভাররাইডিং মেথড বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

৯৯। অ্যাবস্ট্রাক্ট ক্লাস কী?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৭ নং দ্রষ্টব্য।

১০০। ফাইনাল (final) কীওয়ার্ড কী?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৮ নং দ্রষ্টব্য।

১০১। ইনহেরিটেন্স ব্যবহারের সুবিধা কী কী?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৯ নং দ্রষ্টব্য।

১০২। Built-in package গুলো কী কী?
উত্তর সংকেত ঃ অনুশীলনী-৯ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

১০৩। Nested package কী?
উত্তর সংকেত ঃ অনুশীলনী-৯ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

১০৪। জাভা বিনস কী? [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৯ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

১০৫। ইন্টারফেস বলতে কী বুঝায়? এর বৈশিষ্ট্য কী? [বাকাশিবো-২০১২]
উত্তর সংকেত ঃ অনুশীলনী-৯ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৮ নং দ্রষ্টব্য।

১০৬। ইউজার ডিফাই প্যাকেজ কাকে বলে? [বাকাশিবো-২০১৪]
উত্তর সংকেত ঃ অনুশীলনী-৯ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১০ নং দ্রষ্টব্য।

১০৭। জাভা সিস্টেম প্যাকেজ বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-৯ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

১০৮। ইন্টারফেস বাস্তবায়নের ফরম্যাট লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৯ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১২ নং দ্রষ্টব্য।

১০৯। থ্রেড (Thread) কী? [বাকাশিবো-২০১২]
উত্তর সংকেত ঃ অনুশীলনী-১০ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

১১০। সিঙ্গেল থ্রেডেড প্রোগ্রাম কাকে বলে?
উত্তর সংকেত ঃ অনুশীলনী-১০ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

১১১। থ্রেড Priority বলতে কী বোঝায়?
উত্তর সংকেত ঃ অনুশীলনী-১০ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

১১২। Multithread Programming বলতে কী বুঝায়? [বাকাশিবো-২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-১০ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

১১৩। Multitasking কী? [বাকাশিবো-২০১৩, ২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-১০ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

১১৪। Thread synchronization বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-১০ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৭ নং দ্রষ্টব্য।

১১৫। থ্রেড ক্লাস তৈরির প্রক্রিয়াগুলো কী কী? [বাকাশিবো-২০১১, ২০১০, ২০১৪]
উত্তর সংকেত ঃ অনুশীলনী-১০ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৮ নং দ্রষ্টব্য।

১১৬। wait ( ) মেথডের ব্যবহার লেখ।
উত্তর সংকেত ঃ অনুশীলনী-১০ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৯ নং দ্রষ্টব্য।

১১৭। Blocked state কী?
উত্তর সংকেত ঃ অনুশীলনী-১০ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১০ নং দ্রষ্টব্য।

১১৮। থ্রেড সাসপেন্ডিং বলতে কী বুঝায়?
উত্তর সংকেত ঃ অনুশীলনী-১০ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৫ নং দ্রষ্টব্য।

১১৯। থ্রেড রিসিউমিং কী?
উত্তর সংকেত ঃ অনুশীলনী-১০ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১৬ নং দ্রষ্টব্য।

১২০। এনিউমিরেশন (Enumeration) কী?
উত্তর সংকেত ঃ অনুশীলনী-১১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

১২১। এনিউমিরেশন তৈরির ফরম্যাট লেখ।
উত্তর সংকেত ঃ অনুশীলনী-১১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

১২২। Type wrappers কী?
উত্তর সংকেত ঃ অনুশীলনী-১১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

১২৩। Annotations বা Metadata কাকে বলে? [বাকাশিবো-২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-১১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৭ নং দ্রষ্টব্য।

১২৪। অ্যানোটেশন তৈরির ফরম্যাট লেখ। [বাকাশিবো-২০১৩, ২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-১১ এর অতি সংক্ষিপ্ত প্রশ্নোত্তর ৮ নং দ্রষ্টব্য।

সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। গ্রাফিক্স প্রোগ্রামের প্রয়োজনীয়তা কী?
উত্তর সংকেত ঃ অনুশীলনী-১ এর সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

২। ellipse ( ) ও fillellipse ( ) ফাংশনের ব্যবহার উল্লেখ কর।
উত্তর সংকেত ঃ অনুশীলনী-১ এর সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

৩। setcolor ( ) ও setbkcolor ( ) ফাংশনের মধ্যে পার্থক্য লেখ।
উত্তর সংকেত ঃ অনুশীলনী-১ এর সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

৪। drawpoly ( ) ও fillpoly ( ) ফাংশনের মধ্যে পার্থক্য লেখ।
উত্তর সংকেত ঃ অনুশীলনী-১ এর সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

৫। cprintf ( ) ও cscanf ( ) ফাংশনের ব্যবহার উল্লেখ কর।
উত্তর সংকেত ঃ অনুশীলনী-১ এর সংক্ষিপ্ত প্রশ্নোত্তর ৭ নং দ্রষ্টব্য।

৬। আয়তক্ষেত্র অঙ্কনের জন্য একটি সি প্রোগ্রাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-১ এর সংক্ষিপ্ত প্রশ্নোত্তর ১০ নং দ্রষ্টব্য।

৭। একটি 3D Bar অঙ্কনের সি প্রেগ্রাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-১ এর সংক্ষিপ্ত প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

৮। setfillstyle ( ) ও floodfill ( ) ফাংশনের মধ্যে পার্থক্য কী?
উত্তর সংকেত ঃ অনুশীলনী-১ এর সংক্ষিপ্ত প্রশ্নোত্তর ১২ নং দ্রষ্টব্য।

৯। একটি বৃত্ত আঁকার সি প্রোগ্রাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-১ এর সংক্ষিপ্ত প্রশ্নোত্তর ১৩ নং দ্রষ্টব্য।

১০। প্রসিডিউর অরিয়েন্টেড প্রোগ্রামিং এর বৈশিষ্ট্য লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৩ এর সংক্ষিপ্ত প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

১১। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর বৈশিষ্ট্য উল্লেখ কর।
উত্তর সংকেত ঃ অনুশীলনী-৩ এর সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

১২। প্রসিডিউর অরিয়েন্টেড ও অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর মধ্যে পার্থক্য লেখ। [বাকাশিবো-২০১২]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

১৩। ট্রাডিশনাল প্রোগ্রামিং-এর অসুবিধাসমূহ লেখ। [বাকাশিবো-২০১৩, ২০১৪-পরি]
অথবা, ট্রাডিশনাল প্রোগ্রামিং ল্যাঙ্গুয়েজের অসুবিধাগুলো লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৩ এর সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

১৪। উদাহরণসহ ক্লাস ঘোষণার ফরম্যাট বা সিনটেক্স লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৩ এর সংক্ষিপ্ত প্রশ্নোত্তর ৭ নং দ্রষ্টব্য।

১৫। ম্যাসেজ পাসিং প্রক্রিয়া চিত্রসহ বর্ণনা কর। [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর সংক্ষিপ্ত প্রশ্নোত্তর ৯ নং দ্রষ্টব্য।

১৬। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর সুবিধাসমূহ উল্লেখ কর।
উত্তর সংকেত ঃ অনুশীলনী-৩ এর সংক্ষিপ্ত প্রশ্নোত্তর ১০ নং দ্রষ্টব্য।

১৭। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর অসুবিধাগুলো কী কী?
উত্তর সংকেত ঃ অনুশীলনী-৩ এর সংক্ষিপ্ত প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

১৮। সি++ প্রোগ্রামের বেসিক স্ট্রাকচার বর্ণনা কর। [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর সংক্ষিপ্ত প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

১৯। সি++ এ ফাংশন ব্যবহারের বিবেচ্য বিষয়গুলো কী কী? [বাকাশিবো-২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর সংক্ষিপ্ত প্রশ্নোত্তর ৭ নং দ্রষ্টব্য।

২০। বৃত্তের ক্ষেত্রফল নির্ণয়ের জন্য ফাংশন ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ। [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর সংক্ষিপ্ত প্রশ্নোত্তর ৮ নং দ্রষ্টব্য।

২১। স্ট্রাকচার এবং ক্লাস এর মধ্যে পার্থক্য লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর সংক্ষিপ্ত প্রশ্নোত্তর ৯ নং দ্রষ্টব্য।

২২। ক্লাস ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।
অথবা, দেখাও যে, ক্লাসের pulbic member function ব্যতিত private member function কে এক্সেস করা যায় না।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর সংক্ষিপ্ত প্রশ্নোত্তর ১২ নং দ্রষ্টব্য।

২৩। অপারেটর ওভারলোডিং এর সাধারণ নিয়মগুলো লেখ। [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর সংক্ষিপ্ত প্রশ্নোত্তর ১৩ নং দ্রষ্টব্য।

২৪। অপারেটর ওভারলোডিং এর সীমাবদ্ধতা লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর সংক্ষিপ্ত প্রশ্নোত্তর ১৫ নং দ্রষ্টব্য।

২৫। সি++ এ কনস্ট্রাক্টর কেন ব্যবহার করা হয়?
উত্তর সংকেত ঃ অনুশীলনী-৪ এর সংক্ষিপ্ত প্রশ্নোত্তর ১৬ নং দ্রষ্টব্য।

২৬। ফাংশন ওভারলোডিং এর শর্তাবলি কী কী?
অথবা, ফাংশন ওভারলেডিং-এর নিয়ম লেখ। [বাকাশিবো-২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর সংক্ষিপ্ত প্রশ্নোত্তর ১৭ নং দ্রষ্টব্য।

২৭। ডেস্ট্রাক্টর ফাংশন কী? উদাহরণসহ বুঝিয়ে লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর সংক্ষিপ্ত প্রশ্নোত্তর ২০ নং দ্রষ্টব্য।

২৮। ফ্রেন্ড ফাংশন কী? একটি ফ্রেন্ড ফাংশন তৈরি কর। [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর সংক্ষিপ্ত প্রশ্নোত্তর ২১ নং দ্রষ্টব্য।

২৯। হায়ারারকিক্যাল ইনহেরিটেন্স এর ফরম্যাট লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর সংক্ষিপ্ত প্রশ্নোত্তর ২২ নং দ্রষ্টব্য।

৩০। ইনহেরিটেন্সের শ্রেণিবিভাগ দেখাও। [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর সংক্ষিপ্ত প্রশ্নোত্তর ২৪ নং দ্রষ্টব্য।

৩১। সিঙ্গেল ইনহেরিটেন্স ও মাল্টিপল ইনহেরিটেন্স এর মধ্যে পার্থক্য লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর সংক্ষিপ্ত প্রশ্নোত্তর ২৭ নং দ্রষ্টব্য।

৩২। জাভার বৈশিষ্ট্যগুলো কী কী? [বাকাশিবো-২০১৩, ২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৫ এর সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

৩৩। জাভা প্রোগ্রামের স্ট্রাকচার লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৫ এর সংক্ষিপ্ত প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

৩৪। জাভা সি++ এর ইন্টারনেট ভার্সন ব্যাখ্যা কর।
উত্তর সংকেত ঃ অনুশীলনী-৫ এর সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

৩৫। জাভা ও সি++ এর মধ্যে পার্থক্য লেখ। [বাকাশিবো-২০১২]
উত্তর সংকেত ঃ অনুশীলনী-৫ এর সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

৩৬। জাভায় প্রোগ্রাম লেখার নিয়মাবলি কী কী?
উত্তর সংকেত ঃ অনুশীলনী-৫ এর সংক্ষিপ্ত প্রশ্নোত্তর ৭ নং দ্রষ্টব্য।

৩৭। জাভাকে যন্ত্র নিরপেক্ষ প্রোগ্রাম ভাষা বলা হয় কেন?
উত্তর সংকেত ঃ অনুশীলনী-৫ এর সংক্ষিপ্ত প্রশ্নোত্তর ৮ নং দ্রষ্টব্য।

৩৮। জাভা ভার্চুয়াল মেশিন (JVM) ব্যবহার করা হয় কেন? ব্যাখ্যা কর। [বাকাশিবো-২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-৫ এর সংক্ষিপ্ত প্রশ্নোত্তর ৯ নং দ্রষ্টব্য।

৩৯। জাভাকে পূর্ণাঙ্গ অবজেক্ট অরিয়েন্টেড ল্যাংগুয়েজ বলা হয় কেন?
উত্তর সংকেত ঃ অনুশীলনী-৫ এর সংক্ষিপ্ত প্রশ্নোত্তর ১০ নং দ্রষ্টব্য।

৪০। জাভাকে মাল্টিথ্রেডেড প্রোগ্রামিং ল্যাংগুয়েজে বলা হয় কেন?
উত্তর সংকেত ঃ অনুশীলনী-৫ এর সংক্ষিপ্ত প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

৪১। জাভা ডাটা টাইপ হায়ারারকী দেখাও। [বাকাশিবো-২০১২]
উত্তর সংকেত ঃ অনুশীলনী-৬ এর সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

৪২। ইনক্রিমেন্ট ও ডিক্রিমেন্ট অপারেটরের পার্থক্য লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৬ এর সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

৪৩। কন্ডিশনাল অপারেটরের ব্যাখ্যা দাও।
উত্তর সংকেত ঃ অনুশীলনী-৬ এর সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

৪৪। if-else স্টেটমেন্টের ফরম্যাট লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৬ এর সংক্ষিপ্ত প্রশ্নোত্তর ৭ নং দ্রষ্টব্য।

৪৫। else-if স্টেটমেন্টের কারণ ব্যাখ্যা কর।
উত্তর সংকেত ঃ অনুশীলনী-৬ এর সংক্ষিপ্ত প্রশ্নোত্তর ৮ নং দ্রষ্টব্য।

৪৬। switch স্টেটমেন্টের Syntax লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৬ এর সংক্ষিপ্ত প্রশ্নোত্তর ৯ নং দ্রষ্টব্য।

৪৭। while ও do-while লুপ স্টেটমেন্টের মধ্যে পার্থক্য লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৬ এর সংক্ষিপ্ত প্রশ্নোত্তর ১০ নং দ্রষ্টব্য।

৪৮। জাভায় ক্লাস ডিক্লারেশনের ফরম্যাট উদাহরণসহ উল্লেখ কর।
উত্তর সংকেত ঃ অনুশীলনী-৭ এর সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

৪৯। রিটার্ন স্টেটমেন্টের কাজ লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৭ এর সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

৫০। রিটার্ন স্টেটমেন্ট লেখার নিয়মাবলি কী কী?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

৫১। মেথড ব্যবহারের সুবিধাসমূহ উল্লেখ কর। [বাকাশিবো-২০১৪-পরি]
অথবা, মেথড ব্যবহারের সুবিধাগুলো কী কী?
উত্তর সংকেত ঃ অনুশীলনী-৭ এর সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

৫২। কনস্ট্রাক্টর মেথডের বৈশিষ্ট্যগুলো উল্লেখ কর। [বাকাশিবো-২০১২]
উত্তর সংকেত ঃ অনুশীলনী-৭ এর সংক্ষিপ্ত প্রশ্নোত্তর ৮ নং দ্রষ্টব্য।

৫৩। ইন্সটেন্স ভেরিয়েবল হাইডিং প্রক্রিয়া বর্ণনা কর।
উত্তর সংকেত ঃ অনুশীলনী-৭ এর সংক্ষিপ্ত প্রশ্নোত্তর ৯ নং দ্রষ্টব্য।

৫৪। উদাহরণসহ গারবেজ কালেকশন পদ্ধতি ব্যাখ্যা কর। [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৭ এর সংক্ষিপ্ত প্রশ্নোত্তর ১০ নং দ্রষ্টব্য।

৫৫। মাল্টিলেভেল ইনহেরিটেন্স এর ব্লক চিত্র অঙ্কন করে ফরম্যাট লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৮ এর সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

৫৬। হায়ারারকিক্যাল ইনহেরিটেন্স এর ব্লকচিত্রসহ ফরম্যাট লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৮ এর সংক্ষিপ্ত প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

৫৭। সিঙ্গেল ইনহেরিটেন্স ও মাল্টিলেভেল ইনহেরিটেন্স এর মধ্যে পার্থক্য লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৮ এর সংক্ষিপ্ত প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

৫৮। উদাহরণসহ পলিমরফিজম প্রক্রিয়াটি ব্যাখ্যা কর।
উত্তর সংকেত ঃ অনুশীলনী-৮ এর সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

৫৯। প্যাকেজ ডিক্লারেশনের প্রয়োজনীয়তা কী?
উত্তর সংকেত ঃ অনুশীলনী-৯ এর সংক্ষিপ্ত প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

৬০। স্যুইং কী? স্যুইং এর ফিচারগুলোর নাম লেখ। [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৯ এর সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

৬১। প্রসেস বেসড ও থ্রেড বেসড মাল্টি টাস্কিং পদ্ধতির পার্থক্য লেখ। [বাকাশিবো-২০১১, ২০১০]
উত্তর সংকেত ঃ অনুশীলনী-১০ এর সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

৬২। আন্তঃথ্রেড কমিউনিকেশন পদ্ধতি ব্যাখ্যা কর। [বাকাশিবো-২০১১, ২০১৪-পরি]
উত্তর সংকেত ঃ অনুশীলনী-১০ এর সংক্ষিপ্ত প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

৬৩। sleep ( ) ও wait ( ) মেথড এর পার্থক্য লেখ।
উত্তর সংকেত ঃ অনুশীলনী-১০ এর সংক্ষিপ্ত প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

৬৪। notify ( ) ও notify All ( ) মেথড ব্যবহারের পার্থক্য লেখ।
উত্তর সংকেত ঃ অনুশীলনী-১০ এর সংক্ষিপ্ত প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

৬৫। values ( ) ও value of ( ) ব্যবহারের মধ্যে পার্থক্য লেখ। [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-১১ এর সংক্ষিপ্ত প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

➥ *রচনামূলক প্রশ্নোত্তর ঃ*

১। বাংলাদেশের জাতীয় পতাকা অঙ্কনের একটি সি প্রোগ্রাম লেখ। [বাকাশিবো-২০১২, ২০১৩R, ২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-১ এর রচনামূলক প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

২। জাপানের জাতীয় পতাকা অঙ্কনের একটি সি প্রোগ্রাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-১ এর রচনামূলক প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

৩। মাউস পয়েন্টার তৈরি করার জন্য একটি সি ভাষার প্রোগ্রাম লেখ। [বাকাশিবো-২০১২]
উত্তর সংকেত ঃ অনুশীলনী-২ এর রচনামূলক প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

৪। একটি 3D অবজেক্ট তৈরির সি প্রোগ্রাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-২ এর রচনামূলক প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

৫। চিত্রসহ সফটওয়্যার বিবর্তনের বিভিন্ন ধাপসমূহ সংক্ষেপে বর্ণনা কর। [বাকাশিবো-২০১৩]
উত্তর সংকেত ঃ অনুশীলনী-৩ এর রচনামূলক প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

৬। বিভিন্ন প্রকার অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজের বর্ণনা লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৩ এর রচনামূলক প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

৭। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর সুবিধা-অসুবিধা আলোচনা কর।
উত্তর সংকেত ঃ অনুশীলনী-৩ এর রচনামূলক প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

৮। উদাহরণসহ সি++ এর বিভিন্ন প্রকার অপারেটরগুলোর বর্ণনা দাও।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর রচনামূলক প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

৯। ক্লাস ব্যবহার করে তিনটি সংখ্যার মধ্যে বৃহত্তম সংখ্যাটি নির্ণয়ের প্রোগ্রাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর রচনামূলক প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

১০। দ্বিঘাত সমীকরণের মূল (বাস্তব ও অবাস্তব) নির্ণয়ের একটি প্রোগ্রাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর রচনামূলক প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

১১। একটি ধনাত্মক পূর্ণ সংখ্যার ফ্যাক্টরিয়াল মান নির্ণয়ের একটি সি++ প্রোগ্রাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর রচনামূলক প্রশ্নোত্তর ৭ নং দ্রষ্টব্য।

১২। শর্তসাপেক্ষে অসমবাহু ত্রিভুজক্ষেত্রের ক্ষেত্রফল নির্ণয়ের একটি সি++ প্রোগ্রাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর রচনামূলক প্রশ্নোত্তর ৮ নং দ্রষ্টব্য।

১৩। ফাংশন ব্যবহার করে ৫০ পর্যন্ত মৌলিক সংখ্যাগুলো নির্ণয়ের প্রোগ্রাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৪ এর রচনামূলক প্রশ্নোত্তর ৯ নং দ্রষ্টব্য।

১৪। Constructor ও Destructor ফাংশন ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ। [বাকাশিবো- ২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর রচনামূলক প্রশ্নোত্তর ১১ নং দ্রষ্টব্য।

১৫। প্রোগ্রামের মাধ্যমে Copy constructor এর ব্যবহার দেখাও। [বাকাশিবো- ২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর রচনামূলক প্রশ্নোত্তর ১২ নং দ্রষ্টব্য।

১৬। মাল্টিলেভেল ইনহেরিটেন্স ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ। [বাকাশিবো-২০১৩, ২০১৩R, ২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-৪ এর রচনামূলক প্রশ্নোত্তর ১৩ নং দ্রষ্টব্য।

১৭। জাভার বৈশিষ্ট্যগুলো বর্ণনা কর।
উত্তর সংকেত ঃ অনুশীলনী-৫ এর রচনামূলক প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

১৮। জাভা ডেভেলপমেন্ট এনভায়রনমেন্ট এর ধাপসমূহ বর্ণনা কর।
উত্তর সংকেত ঃ অনুশীলনী-৫ এর রচনামূলক প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

১৯। জাভা প্রোগ্রামের স্ট্রাকচার বা সংগঠন বর্ণনা কর।
উত্তর সংকেত ঃ অনুশীলনী-৫ এর রচনামূলক প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

২০। 1 + 2 + 3 + ... + N = ? সিজিরে যোগফল নির্ণয়ের জাভা প্রোগ্রাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৬ এর রচনামূলক প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

২১। প্রথম ১০টি সংখ্যার Fibonacci সিরিজ প্রদর্শনের জাভা প্রোগ্রাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৬ এর রচনামূলক প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

২২। 25 পর্যন্ত সংখ্যাগুলোর মধ্যকার মৌলিক সংখ্যাগুলো নির্ণয়ের জাভা প্রোগ্রাম লেখ। [বাকাশিবো-২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-৬ এর রচনামূলক প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

২৩। ক্লাসে মেথড সংযোজনের প্রক্রিয়াটি প্রোগ্রামের মাধ্যমে দেখাও।
উত্তর সংকেত ঃ অনুশীলনী-৭ এর রচনামূলক প্রশ্নোত্তর ৪ নং দ্রষ্টব্য।

২৪। মেথড ওভারলোডিং পদ্ধতির জন্য একটি জাভা প্রোগ্রাম লেখ। [বাকাশিবো-২০১২, ২০১৩R, ২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-৭ এর রচনামূলক প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

২৫। কনস্ট্রাক্টর মেথড ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।
উত্তর সংকেত ঃ অনুশীলনী-৭ এর রচনামূলক প্রশ্নোত্তর ৬ নং দ্রষ্টব্য।

২৬। প্রোগ্রামের মাধ্যমে ওভারলোডিং কনস্ট্রাক্টর বা কপি কনস্ট্রাক্টর পদ্ধতি ব্যাখ্যা কর।
উত্তর সংকেত ঃ অনুশীলনী-৭ এর রচনামূলক প্রশ্নোত্তর ৭ নং দ্রষ্টব্য।

২৭। সুপার ক্লাস ও সাব ক্লাস ব্যবহার করে একটি প্রোগ্রাম লেখ। [বাকাশিবো-২০১২, ২০১৩, ২০১৪R]
উত্তর সংকেত ঃ অনুশীলনী-৮ এর রচনামূলক প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

২৮। অ্যাবস্ট্রাক্ট ক্লাস ব্যবহার করে একটি প্রোগ্রাম লেখ। [বাকাশিবো-২০১৩R]
উত্তর সংকেত ঃ অনুশীলনী-৮ এর রচনামূলক প্রশ্নোত্তর ৫ নং দ্রষ্টব্য।

২৯। ইন্টারফেস বাস্তবায়ন প্রক্রিয়াটি একটি প্রোগ্রাম লিখে ব্যাখ্যা কর।
উত্তর সংকেত ঃ অনুশীলনী-৯ এর রচনামূলক প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

৩০। মাল্টিপল ইন্টারফেস প্রক্রিয়াটি প্রোগ্রামসহ বর্ণনা কর।
উত্তর সংকেত ঃ অনুশীলনী-৯ এর রচনামূলক প্রশ্নোত্তর ৩ নং দ্রষ্টব্য।

৩১। থ্রেডের (thread) জীবন চক্র বর্ণনা কর। [বাকাশিবো–২০১১]
অথবা, স্টেট ডায়াগ্রামসহ থ্রেড ক্লাসের মেথডসমূহ বর্ণনা কর। [বাকাশিবো-২০০১২]
উত্তর সংকেত ঃ অনুশীলনী-১০ এর রচনামূলক প্রশ্নোত্তর ১ নং দ্রষ্টব্য।

৩২। থ্রেড কন্ট্রোল মেথডগুলো বর্ণনা কর।
উত্তর সংকেত ঃ অনুশীলনী-১০ এর রচনামূলক প্রশ্নোত্তর ২ নং দ্রষ্টব্য।

বাংলাদেশ কারিগরি শিক্ষা বোর্ড, ঢাকা

ডিপ্লোমা-ইন-ইঞ্জিনিয়ারিং

৪র্থ পর্ব সমাপনী পরীক্ষা -২০১২

টেকনোলজি ঃ কম্পিউটার, ডাটা টেলিকমিউনিকেশন অ্যান্ড নেটওয়ার্কিং ও কম্পিউটার সাইয়েন্স অ্যান্ড টেকনোলজি

(২০১০ প্রবিধান)

বিষয় ঃ প্রোগ্রামিং ল্যাংগুয়েজ-২

(বিষয় কোড ঃ ৬৬৪২)

সময় ঃ ৩ ঘণ্টা **পূর্ণমান ঃ ৮০**

ক ও খ বিভাগের সকল প্রশ্নের এবং গ-বিভাগের যে-কোন ৫(পাঁচ) টি প্রশ্নের উত্তর দাও।

ক-বিভাগ (মান ঃ ২ × ১০ = ২০)

১। Resolution বলতে কী বোঝায়?

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

২। Software ইন্টারাপ্ট কী?

উত্তর সংকেত ঃ অনুশীলনী ২ এর অতি সংক্ষিপ্ত ৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

৩। Inheritance কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৯ নং প্রশ্নোত্তর দ্রষ্টব্য।

৪। Copy construction কী?

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৩২ নং প্রশ্নোত্তর দ্রষ্টব্য।

৫। Literals কী?

উত্তর সংকেত ঃ অনুশীলনী ৫ এর অতি সংক্ষিপ্ত ১২ নং প্রশ্নোত্তর দ্রষ্টব্য।

৬। Java কী?

উত্তর ঃ জাভা হচ্ছে একটি সত্যিকারের পূর্ণাঙ্গ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ।

৭। Thread কী?

উত্তর সংকেত ঃ অনুশীলনী ১০ এর অতি সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

৮। Data abstraction কী?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ১০ নং প্রশ্নোত্তর দ্রষ্টব্য।

৯। Package বলতে কী বোঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর অতি সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১০। Swing কী?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর সংক্ষিপ্ত ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

খ-বিভাগ (মান ঃ ৩ × ১০ = ৩০)

১১। Initgraph () ফাংশনের কাজ কী, বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১২। OOP এবং POP এর মাঝে পার্থক্য লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর সংক্ষিপ্ত ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

বাংলাদেশ কারিগরি শিক্ষা বোর্ড, ঢাকা

ডিপ্লোমা-ইন-ইঞ্জিনিয়ারিং

৪র্থ পর্ব সমাপনী পরীক্ষা -২০১২

টেকনোলজি ঃ কম্পিউটার, ডাটা টেলিকমিউনিকেশন অ্যান্ড নেটওয়ার্কিং ও কম্পিউটার সাইয়েন্স অ্যান্ড টেকনোলজি

(২০১০ প্রবিধান)

বিষয় ঃ প্রোগ্রামিং ল্যাংগুয়েজ-২

(বিষয় কোড ঃ ৬৬৪২)

সময় ঃ ৩ ঘণ্টা **পূর্ণমান ঃ ৮০**

ক ও খ বিভাগের সকল প্রশ্নের এবং গ-বিভাগের যে-কোন ৫(পাঁচ) টি প্রশ্নের উত্তর দাও।

ক-বিভাগ (মান ঃ ২ × ১০ = ২০)

১। Resolution বলতে কী বোঝায়?

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

২। Software ইন্টারাপ্ট কী?

উত্তর সংকেত ঃ অনুশীলনী ২ এর অতি সংক্ষিপ্ত ৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

৩। Inheritance কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৯ নং প্রশ্নোত্তর দ্রষ্টব্য।

৪। Copy construction কী?

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৩২ নং প্রশ্নোত্তর দ্রষ্টব্য।

৫। Literals কী?

উত্তর সংকেত ঃ অনুশীলনী ৫ এর অতি সংক্ষিপ্ত ১২ নং প্রশ্নোত্তর দ্রষ্টব্য।

৬। Java কী?

উত্তর ঃ জাভা হচ্ছে একটি সত্যিকারের পূর্ণাঙ্গ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ।

৭। Thread কী?

উত্তর সংকেত ঃ অনুশীলনী ১০ এর অতি সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

৮। Data abstraction কী?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ১০ নং প্রশ্নোত্তর দ্রষ্টব্য।

৯। Package বলতে কী বোঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর অতি সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১০। Swing কী?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর সংক্ষিপ্ত ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

খ-বিভাগ (মান ঃ ৩ × ১০ = ৩০)

১১। Initgraph () ফাংশনের কাজ কী, বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১২। OOP এবং POP এর মাঝে পার্থক্য লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর সংক্ষিপ্ত ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

বাংলাদেশ কারিগরি শিক্ষা বোর্ড, ঢাকা

ডিপ্লোমা-ইন-ইঞ্জিনিয়ারিং

৪র্থ পর্ব সমাপনী পরীক্ষা -২০১২

টেকনোলজি ঃ কম্পিউটার, ডাটা টেলিকমিউনিকেশন অ্যান্ড নেটওয়ার্কিং ও কম্পিউটার সাইয়েন্স অ্যান্ড টেকনোলজি

(২০১০ প্রবিধান)

বিষয় ঃ প্রোগ্রামিং ল্যাংগুয়েজ-২

(বিষয় কোড ঃ ৬৬৪২)

সময় ঃ ৩ ঘণ্টা **পূর্ণমান ঃ ৮০**

ক ও খ বিভাগের সকল প্রশ্নের এবং গ-বিভাগের যে-কোন ৫(পাঁচ) টি প্রশ্নের উত্তর দাও।

ক-বিভাগ (মান ঃ ২ × ১০ = ২০)

১। Resolution বলতে কী বোঝায়?

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

২। Software ইন্টারাপ্ট কী?

উত্তর সংকেত ঃ অনুশীলনী ২ এর অতি সংক্ষিপ্ত ৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

৩। Inheritance কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৯ নং প্রশ্নোত্তর দ্রষ্টব্য।

৪। Copy construction কী?

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৩২ নং প্রশ্নোত্তর দ্রষ্টব্য।

৫। Literals কী?

উত্তর সংকেত ঃ অনুশীলনী ৫ এর অতি সংক্ষিপ্ত ১২ নং প্রশ্নোত্তর দ্রষ্টব্য।

৬। Java কী?

উত্তর ঃ জাভা হচ্ছে একটি সত্যিকারের পূর্ণাঙ্গ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ।

৭। Thread কী?

উত্তর সংকেত ঃ অনুশীলনী ১০ এর অতি সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

৮। Data abstraction কী?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ১০ নং প্রশ্নোত্তর দ্রষ্টব্য।

৯। Package বলতে কী বোঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর অতি সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১০। Swing কী?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর সংক্ষিপ্ত ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

খ-বিভাগ (মান ঃ ৩ × ১০ = ৩০)

১১। Initgraph () ফাংশনের কাজ কী, বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১২। OOP এবং POP এর মাঝে পার্থক্য লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর সংক্ষিপ্ত ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

বাংলাদেশ কারিগরি শিক্ষা বোর্ড, ঢাকা

ডিপ্লোমা-ইন-ইঞ্জিনিয়ারিং

৪র্থ পর্ব সমাপনী পরীক্ষা -২০১২

টেকনোলজি ঃ কম্পিউটার, ডাটা টেলিকমিউনিকেশন অ্যান্ড নেটওয়ার্কিং ও কম্পিউটার সাইয়েন্স অ্যান্ড টেকনোলজি

(২০১০ প্রবিধান)

বিষয় ঃ প্রোগ্রামিং ল্যাংগুয়েজ-২

(বিষয় কোড ঃ ৬৬৪২)

সময় ঃ ৩ ঘণ্টা **পূর্ণমান ঃ ৮০**

ক ও খ বিভাগের সকল প্রশ্নের এবং গ-বিভাগের যে-কোন ৫(পাঁচ) টি প্রশ্নের উত্তর দাও।

ক-বিভাগ (মান ঃ ২ × ১০ = ২০)

১। Resolution বলতে কী বোঝায়?

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

২। Software ইন্টারাপ্ট কী?

উত্তর সংকেত ঃ অনুশীলনী ২ এর অতি সংক্ষিপ্ত ৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

৩। Inheritance কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৯ নং প্রশ্নোত্তর দ্রষ্টব্য।

৪। Copy construction কী?

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৩২ নং প্রশ্নোত্তর দ্রষ্টব্য।

৫। Literals কী?

উত্তর সংকেত ঃ অনুশীলনী ৫ এর অতি সংক্ষিপ্ত ১২ নং প্রশ্নোত্তর দ্রষ্টব্য।

৬। Java কী?

উত্তর ঃ জাভা হচ্ছে একটি সত্যিকারের পূর্ণাঙ্গ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ।

৭। Thread কী?

উত্তর সংকেত ঃ অনুশীলনী ১০ এর অতি সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

৮। Data abstraction কী?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ১০ নং প্রশ্নোত্তর দ্রষ্টব্য।

৯। Package বলতে কী বোঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর অতি সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১০। Swing কী?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর সংক্ষিপ্ত ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

খ-বিভাগ (মান ঃ ৩ × ১০ = ৩০)

১১। Initgraph () ফাংশনের কাজ কী, বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১২। OOP এবং POP এর মাঝে পার্থক্য লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর সংক্ষিপ্ত ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

বাংলাদেশ কারিগরি শিক্ষা বোর্ড, ঢাকা

ডিপ্লোমা-ইন-ইঞ্জিনিয়ারিং

৪র্থ পর্ব সমাপনী পরীক্ষা -২০১২

টেকনোলজি ঃ কম্পিউটার, ডাটা টেলিকমিউনিকেশন অ্যান্ড নেটওয়ার্কিং ও কম্পিউটার সাইয়েন্স অ্যান্ড টেকনোলজি

(২০১০ প্রবিধান)

বিষয় ঃ প্রোগ্রামিং ল্যাংগুয়েজ-২

(বিষয় কোড ঃ ৬৬৪২)

সময় ঃ ৩ ঘণ্টা **পূর্ণমান ঃ ৮০**

ক ও খ বিভাগের সকল প্রশ্নের এবং গ-বিভাগের যে-কোন ৫(পাঁচ) টি প্রশ্নের উত্তর দাও।

ক-বিভাগ (মান ঃ ২ × ১০ = ২০)

১। Resolution বলতে কী বোঝায়?

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

২। Software ইন্টারাপ্ট কী?

উত্তর সংকেত ঃ অনুশীলনী ২ এর অতি সংক্ষিপ্ত ৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

৩। Inheritance কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৯ নং প্রশ্নোত্তর দ্রষ্টব্য।

৪। Copy construction কী?

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৩২ নং প্রশ্নোত্তর দ্রষ্টব্য।

৫। Literals কী?

উত্তর সংকেত ঃ অনুশীলনী ৫ এর অতি সংক্ষিপ্ত ১২ নং প্রশ্নোত্তর দ্রষ্টব্য।

৬। Java কী?

উত্তর ঃ জাভা হচ্ছে একটি সত্যিকারের পূর্ণাঙ্গ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ।

৭। Thread কী?

উত্তর সংকেত ঃ অনুশীলনী ১০ এর অতি সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

৮। Data abstraction কী?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ১০ নং প্রশ্নোত্তর দ্রষ্টব্য।

৯। Package বলতে কী বোঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর অতি সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১০। Swing কী?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর সংক্ষিপ্ত ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

খ-বিভাগ (মান ঃ ৩ × ১০ = ৩০)

১১। Initgraph () ফাংশনের কাজ কী, বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১২। OOP এবং POP এর মাঝে পার্থক্য লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর সংক্ষিপ্ত ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৩। Student নামের একটি নমুনা ক্লাসের ফরম্যাট লেখ।

উত্তর সংকেত ঃ ৩.৩ নং অনুচ্ছেদ (Class দ্রষ্টব্য এর Program)।

১৪। C^{++} এ I/O Stream বর্ণনা কর।

উত্তর সংকেত ঃ ৪.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

১৫। Constructor funciton এর বৈশিষ্ট্যগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর সংক্ষিপ্ত ৮ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৬। Java তে Inheritance কয় প্রকার ও কী কী?

উত্তর সংকেত ঃ অনুশীলনী ৭ এর অতি সংক্ষিপ্ত ২ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৭। তিনটি সংখ্যার মাঝে ছোট সংখ্যা বের করার জন্য C^{++} এ একটি Program লেখ।

উত্তর সংকেত ঃ নিজে চেষ্টা কর।

১৮। জাভায় ব্যবহৃত ডাটা টাইপের একটি হায়ারারকি দেখাও।

উত্তর সংকেত ঃ অনুশীলনী ৬ এর সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৯। ইন্টারফেস বলতে কী বোঝায়? এর বৈশিষ্ট্য কী?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর অতি সংক্ষিপ্ত ৮ নং প্রশ্নোত্তর দ্রষ্টব্য।

২০। Java ও C^{++} এর মাঝে পার্থক্য লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৫ এর সংক্ষিপ্ত ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

গ-বিভাগ (মান ঃ ৬ × ৫ = ৩০)

২১। Class ও Object ব্যবহার করে একটি C^{++} প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ৩.৩ নং অনুচ্ছেদ (Inhenitance প্রোগ্রাম) দ্রষ্টব্য।

২২। বাংলাদেশের পতাকা অঙ্কনের একটি C প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ২ এর রচনামূলক ২ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৩। সুপার ক্লাস ও সাব-ক্লাস ব্যবহার করে একটি Java প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর রচনামূলক ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৪। একটি 'Array তে' দশটি বিষয়ের মার্ক কী-বোর্ড থেকে ইনপুট নিয়ে এর গড় নির্ণয়ের জন্য C^{++} এ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুরূপ (সি$^{++}$ প্রোগ্রামসমূহে) প্রোগ্রাম-১.৫ দ্রষ্টব্য।

২৫। স্টেট ডায়াগ্রামসহ থ্রেড ক্লাসের বিভিন্ন মেথডগুলো বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ১০ এর রচনামূলক ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৬। Method ওভারলোডিং এর একটি Java প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর রচনামূলক ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৩। Student নামের একটি নমুনা ক্লাসের ফরম্যাট লেখ।

উত্তর সংকেত ঃ ৩.৩ নং অনুচ্ছেদ (Class দ্রষ্টব্য এর Program)।

১৪। C^{++} এ I/O Stream বর্ণনা কর।

উত্তর সংকেত ঃ ৪.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

১৫। Constructor funciton এর বৈশিষ্ট্যগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর সংক্ষিপ্ত ৮ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৬। Java তে Inheritance কয় প্রকার ও কী কী?

উত্তর সংকেত ঃ অনুশীলনী ৭ এর অতি সংক্ষিপ্ত ২ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৭। তিনটি সংখ্যার মাঝে ছোট সংখ্যা বের করার জন্য C^{++} এ একটি Program লেখ।

উত্তর সংকেত ঃ নিজে চেষ্টা কর।

১৮। জাভায় ব্যবহৃত ডাটা টাইপের একটি হায়ারারকি দেখাও।

উত্তর সংকেত ঃ অনুশীলনী ৬ এর সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৯। ইন্টারফেস বলতে কী বোঝায়? এর বৈশিষ্ট্য কী?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর অতি সংক্ষিপ্ত ৮ নং প্রশ্নোত্তর দ্রষ্টব্য।

২০। Java ও C^{++} এর মাঝে পার্থক্য লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৫ এর সংক্ষিপ্ত ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

গ-বিভাগ (মান ঃ ৬ × ৫ = ৩০)

২১। Class ও Object ব্যবহার করে একটি C^{++} প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ৩.৩ নং অনুচ্ছেদ (Inhenitance প্রোগ্রাম) দ্রষ্টব্য।

২২। বাংলাদেশের পতাকা অঙ্কনের একটি C প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ২ এর রচনামূলক ২ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৩। সুপার ক্লাস ও সাব-ক্লাস ব্যবহার করে একটি Java প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর রচনামূলক ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৪। একটি 'Array তে' দশটি বিষয়ের মার্ক কী-বোর্ড থেকে ইনপুট নিয়ে এর গড় নির্ণয়ের জন্য C^{++} এ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুরূপ (সি$^{++}$ প্রোগ্রামসমূহে) প্রোগ্রাম-১.৫ দ্রষ্টব্য।

২৫। স্টেট ডায়াগ্রামসহ থ্রেড ক্লাসের বিভিন্ন মেথডগুলো বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ১০ এর রচনামূলক ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৬। Method ওভারলোডিং এর একটি Java প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর রচনামূলক ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

বাংলাদেশ কারিগরি শিক্ষা বোর্ড, ঢাকা

ডিপ্লোমা-ইন-ইঞ্জিনিয়ারিং

৪র্থ পর্ব সমাপনী পরীক্ষা-২০১৩

টেকনোলজি ঃ কম্পিউটার, কম্পিউটার সায়েন্স ও ডাটা টেলিকমিউনিকেশন (২০১০ প্রবিধান)

বিষয় ঃ প্রোগ্রামিং ল্যাংগুয়েজ-২

বিষয় কোড ঃ ৬৬৪২

সময় ঃ ৩ ঘণ্টা পূর্ণমান ঃ ৮০

ক ও খ-বিভাগের **সকল** প্রশ্নের এবং গ-বিভাগের যে-কোন ৫ (পাঁচ) টি প্রশ্নের উত্তর দাও।

ক-বিভাগ (মান ঃ ২ × ১০ = ২০)

১। OOP বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

২। গ্রাফিক্স কার্ডের রেজোলুশন 1024 × 768 বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ১১ নং প্রশ্নোত্তর দ্রষ্টব্য।

৩। ইন্টারাপ্ট কী?

উত্তর সংকেত ঃ অনুশীলনী ২ এর অতি সংক্ষিপ্ত ২ নং প্রশ্নোত্তর দ্রষ্টব্য।

৪। অবজেক্ট বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৭ নং প্রশ্নোত্তর দ্রষ্টব্য।

৫। কনস্ট্রাক্টর কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ৭ এর অতি সংক্ষিপ্ত ১০ নং প্রশ্নোত্তর দ্রষ্টব্য।

৬। আইডেন্টিফায়ার কী?

উত্তর সংকেত ঃ অনুশীলনী ৫ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

৭। ডাইনামিক ইনিশিয়ালাইজেশন কী?

উত্তর সংকেত ঃ অনুশীলনী ৬ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

৮। প্যাকেজ ঘোষণার সাধারণ ফরমেট লেখ।

উত্তর ঃ প্যাকেজ ঘোষণার সাধারণ ফরফেট হল– Package package Name।

৯। অ্যানোটেশন তৈরির ফরমেট লেখ।

উত্তর সংকেত ঃ অনুশীলনী ১১ এর অতি সংক্ষিপ্ত ৮ নং প্রশ্নোত্তর দ্রষ্টব্য।

১০। মাল্টিটাস্কিং বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ১০ এর অতি সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

খ-বিভাগ (মান ঃ ৩ × ১০ = ৩০)

১১। একটি বৃত্তের ক্ষেত্রফল নির্ণয়ের প্রোগ্রাম C++ ভাষায় লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর সংক্ষিপ্ত ৮ নং প্রশ্নোত্তর দ্রষ্টব্য।

১২। ট্রাডিশনাল প্রোগ্রামিং এর অসুবিধাগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

বাংলাদেশ কারিগরি শিক্ষা বোর্ড, ঢাকা

ডিপ্লোমা-ইন-ইঞ্জিনিয়ারিং

৪র্থ পর্ব সমাপনী পরীক্ষা-২০১৩

টেকনোলজি ঃ কম্পিউটার, কম্পিউটার সায়েন্স ও ডাটা টেলিকমিউনিকেশন (২০১০ প্রবিধান)

বিষয় ঃ প্রোগ্রামিং ল্যাংগুয়েজ-২

বিষয় কোড ঃ ৬৬৪২

সময় ঃ ৩ ঘণ্টা পূর্ণমান ঃ ৮০

ক ও খ-বিভাগের **সকল** প্রশ্নের এবং গ-বিভাগের যে-কোন ৫ (পাঁচ) টি প্রশ্নের উত্তর দাও।

ক-বিভাগ (মান ঃ ২ × ১০ = ২০)

১। OOP বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

২। গ্রাফিক্স কার্ডের রেজোলুশন 1024 × 768 বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ১১ নং প্রশ্নোত্তর দ্রষ্টব্য।

৩। ইন্টারাপ্ট কী?

উত্তর সংকেত ঃ অনুশীলনী ২ এর অতি সংক্ষিপ্ত ২ নং প্রশ্নোত্তর দ্রষ্টব্য।

৪। অবজেক্ট বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৭ নং প্রশ্নোত্তর দ্রষ্টব্য।

৫। কনস্ট্রাক্টর কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ৭ এর অতি সংক্ষিপ্ত ১০ নং প্রশ্নোত্তর দ্রষ্টব্য।

৬। আইডেন্টিফায়ার কী?

উত্তর সংকেত ঃ অনুশীলনী ৫ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

৭। ডাইনামিক ইনিশিয়ালাইজেশন কী?

উত্তর সংকেত ঃ অনুশীলনী ৬ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

৮। প্যাকেজ ঘোষণার সাধারণ ফরমেট লেখ।

উত্তর ঃ প্যাকেজ ঘোষণার সাধারণ ফরফেট হল– Package package Name।

৯। অ্যানোটেশন তৈরির ফরমেট লেখ।

উত্তর সংকেত ঃ অনুশীলনী ১১ এর অতি সংক্ষিপ্ত ৮ নং প্রশ্নোত্তর দ্রষ্টব্য।

১০। মাল্টিটাস্কিং বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ১০ এর অতি সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

খ-বিভাগ (মান ঃ ৩ × ১০ = ৩০)

১১। একটি বৃত্তের ক্ষেত্রফল নির্ণয়ের প্রোগ্রাম C++ ভাষায় লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর সংক্ষিপ্ত ৮ নং প্রশ্নোত্তর দ্রষ্টব্য।

১২। ট্রাডিশনাল প্রোগ্রামিং এর অসুবিধাগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

বাংলাদেশ কারিগরি শিক্ষা বোর্ড, ঢাকা

ডিপ্লোমা-ইন-ইঞ্জিনিয়ারিং

৪র্থ পর্ব সমাপনী পরীক্ষা-২০১৩

টেকনোলজি ঃ কম্পিউটার, কম্পিউটার সায়েন্স ও ডাটা টেলিকমিউনিকেশন (২০১০ প্রবিধান)

বিষয় ঃ প্রোগ্রামিং ল্যাংগুয়েজ-২

বিষয় কোড ঃ ৬৬৪২

সময় ঃ ৩ ঘণ্টা পূর্ণমান ঃ ৮০

ক ও খ-বিভাগের **সকল** প্রশ্নের এবং গ-বিভাগের যে-কোন ৫ (পাঁচ) টি প্রশ্নের উত্তর দাও।

ক-বিভাগ (মান ঃ ২ × ১০ = ২০)

১। OOP বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

২। গ্রাফিক্স কার্ডের রেজোলুশন 1024 × 768 বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ১১ নং প্রশ্নোত্তর দ্রষ্টব্য।

৩। ইন্টারাপ্ট কী?

উত্তর সংকেত ঃ অনুশীলনী ২ এর অতি সংক্ষিপ্ত ২ নং প্রশ্নোত্তর দ্রষ্টব্য।

৪। অবজেক্ট বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৭ নং প্রশ্নোত্তর দ্রষ্টব্য।

৫। কনস্ট্রাক্টর কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ৭ এর অতি সংক্ষিপ্ত ১০ নং প্রশ্নোত্তর দ্রষ্টব্য।

৬। আইডেন্টিফায়ার কী?

উত্তর সংকেত ঃ অনুশীলনী ৫ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

৭। ডাইনামিক ইনিশিয়ালাইজেশন কী?

উত্তর সংকেত ঃ অনুশীলনী ৬ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

৮। প্যাকেজ ঘোষণার সাধারণ ফরমেট লেখ।

উত্তর ঃ প্যাকেজ ঘোষণার সাধারণ ফরফেট হল– Package package Name।

৯। অ্যানোটেশন তৈরির ফরমেট লেখ।

উত্তর সংকেত ঃ অনুশীলনী ১১ এর অতি সংক্ষিপ্ত ৮ নং প্রশ্নোত্তর দ্রষ্টব্য।

১০। মাল্টিটাস্কিং বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ১০ এর অতি সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

খ-বিভাগ (মান ঃ ৩ × ১০ = ৩০)

১১। একটি বৃত্তের ক্ষেত্রফল নির্ণয়ের প্রোগ্রাম C++ ভাষায় লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর সংক্ষিপ্ত ৮ নং প্রশ্নোত্তর দ্রষ্টব্য।

১২। ট্রাডিশনাল প্রোগ্রামিং এর অসুবিধাগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৩। উদাহরণসহ একটি পূর্ণাঙ্গ ক্লাসের গঠন দেখাও।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর সংক্ষিপ্ত ২ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৪। ম্যাসেজ পাসিং প্রক্রিয়া চিত্রসহ বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর সংক্ষিপ্ত ৯ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৫। অপারেটর ওভারলোডিং-এর উদাহরণসহ বর্ণনা দাও।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৩০ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৬। Inline ফাংশন বলতে কী বুঝায়? উদাহরণসহ লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ২১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৭। 1, 2, 3, 5, 8, 13, 21 সিরিজটি নির্ণয়ের জন্য C++ ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ নিজে চেষ্টা কর।

১৮। জাভা ভাষার বৈশিষ্ট্যগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৫ এর সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৯। 1 + 3 + 5 + 7 + ... + 99 সিরিজটির মান নির্ণয়ের প্রোগ্রাম C++ ভাষায় রচনা কর।

উত্তর সংকেত ঃ নিজে চেষ্টা কর।

২০। ইনহেরিটেন্স কাকে বলে? এটির প্রকারভেদ লেখ।

উত্তর সংকেত ঃ ৪.৮ নং অনুচ্ছেদ দ্রষ্টব্য।

গ-বিভাগ (মান ঃ ৬ × ৫ = ৩০)

২১। মাল্টিলেভেল ইনহেরিটেন্স ব্যবহার করে C++ ভাষায় প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর রচনামূলক ১৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

২২। কপি কনস্ট্রাক্টর ব্যবহার করে Java ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ব্যবহারিক (জাভা প্রোগ্রামসমূহ) প্রোগ্রাম-১২ দ্রষ্টব্য।

২৩। 1 – 100 পর্যন্ত মৌলিক সংখ্যাগুলো নির্ণয়ের জন্য C++ ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ব্যবহারিক C++ (Pragram-11) দ্রষ্টব্য।

২৪। সুপার-ক্লাস ও সাব-ক্লাস ব্যবহার করে C++ ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর রচনামূলক ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৫। কী-বোর্ড থেকে একটি সংখ্যা নিয়ে তার গ্রেড নির্ণয় করার জন্য C ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুরূপ ব্যবহারিক (C প্রোগ্রামসমূহ) প্রোগ্রাম-১৫ দ্রষ্টব্য।

২৬। সফটওয়্যার বিবর্তনের ধাপগুলো বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর রচনামূলক ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

বাংলাদেশ কারিগরি শিক্ষা বোর্ড, ঢাকা

ডিপ্লোমা-ইন-ইঞ্জিনিয়ারিং

৪র্থ পর্ব পরিপূরক পরীক্ষা-২০১৩

টেকনোলজি ঃ কম্পিউটার, কম্পিউটার সাইয়েন্স অ্যান্ড টেকনোলজি ও ডাটা টেলিকমিউনিকেশন অ্যান্ড নেটওয়ার্কিং

(২০১০ প্রবিধান)

বিষয় ঃ প্রোগ্রামিং ল্যাংগুয়েজ-২

বিষয় কোড ঃ ৬৬৪২

সময় ঃ ৩ ঘণ্টা পূর্ণমান ঃ ৮০

ক ও খ-বিভাগের **সকল** প্রশ্নের এবং গ-বিভাগের যে-কোন ৫ **(পাঁচ)** টি প্রশ্নের উত্তর দাও।

ক-বিভাগ (মান ঃ ২ × ১০ = ২০)

১। অবজেক্ট অরিয়েন্টেড প্রোগ্রাম বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

২। ইনহেরিটেন্স কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৯ নং প্রশ্নোত্তর দ্রষ্টব্য।

৩। রেজোলুশন বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

৪। ক্লাশ মেম্বার অ্যাক্সেসিবিলিটি মোডগুলোর নাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

৫। কপি কনস্ট্রাক্টর বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৩২ নং প্রশ্নোত্তর দ্রষ্টব্য।

৬। Class বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৮ নং প্রশ্নোত্তর দ্রষ্টব্য।

৭। প্যাকেজ বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর অতি সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

৮। মেথড কী? মেথডের উপাদানগুলোর নাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর অতি সংক্ষিপ্ত ৫ ও ৭ নং প্রশ্নোত্তর দ্রষ্টব্য।

৯। Initgraph ( ) ফাংশনের কাজ কী?

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১০। জাভা বিন বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর অতি সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

খ-বিভাগ (মান ঃ ৩ × ১০ = ৩০)

১১। জাভা প্রোগ্রামের বৈশিষ্ট্যগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৫ এর সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১২। গার্বেজ কালেকশন কী, বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর সংক্ষিপ্ত ১০ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৩। মাউস প্রোগ্রামিং বলতে কী বুঝায়? এটির বিবেচ্য বিষয়গুলো কী কী?

উত্তর সংকেত ঃ অনুশীলনী ২ এর অতি সংক্ষিপ্ত ৫ ও ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৪। ইন-লাইন ফাংশন বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ২১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৫। অপারেটর ওভারলোডিং কী? এটির সাধারণ নিয়মগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৩০ এবং সংক্ষিপ্ত ১৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৬। সি++ প্রোগ্রামের ইনহেরিটেন্সগুলোর ফরম্যাট লেখ।

উত্তর সংকেত ঃ ৪.৮ নং অনুচ্ছেদ (Inhreritance mode) দ্রষ্টব্য।

১৭। কনসোল I/O কী? ফরম্যাটেড অপারেশনের প্রয়োজনীয়তা কী?

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৫ ও ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৮। Friend Function কী? এটির ফরম্যাট লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর সংক্ষিপ্ত ২১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৯। Values ( ) এবং Values of ( ) Method দুটির পার্থক্য লেখ।

উত্তর সংকেত ঃ অনুশীলনী ১১ এর সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

২০। তিনটি সংখ্যার মাঝে ছোট সংখ্যা বের করার জন্য C++ এ Program লেখ।

উত্তর সংকেত ঃ নিজে চেষ্টা কর।

গ-বিভাগ (মান ঃ ৬ × ৫ = ৩০)

২১। কপি কনস্ট্রাক্টর ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর রচনামূলক ১২ নং প্রশ্নোত্তর দ্রষ্টব্য।

২২। বাংলাদেশের জাতীয় পতাকা অঙ্কনের একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ১ এর রচনামূলক ২ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৩। মেথড ওভারলোডিং ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর রচনামূলক ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৪। মাল্টিলেভেল ইনহেরিটেন্স ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর রচনামূলক ১৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৫। ক্লাশ ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর রচনামূলক ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৬। কনস্ট্রাক্টর ও ডিস্ট্রাক্টর ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর রচনামূলক ১১ নং প্রশ্নোত্তর দ্রষ্টব্য।

বাংলাদেশ কারিগরি শিক্ষা বোর্ড, ঢাকা

ডিপ্লোমা-ইন-ইঞ্জিনিয়ারিং

৪র্থ পর্ব পরিপূরক পরীক্ষা-২০১৪ (পরীক্ষার তারিখ ঃ ২/৭/২০১৪)

টেকনোলজি ঃ কম্পিউটার, ডাটা টেলিকমিউনিকেশন অ্যান্ড নেটওয়ার্ক ও কম্পিউটার সায়েন্স (২০১০ প্রবিধান)

বিষয় ঃ প্রোগ্রামিং ল্যাংগুয়েজ-২

বিষয় কোড ঃ ৬৬৪২

সময় ঃ ৩ ঘণ্টা পূর্ণমান ঃ ৮০

ক ও খ-বিভাগের **সকল** প্রশ্নের এবং গ-বিভাগের যে-কোন ৫ (পাঁচ) টি প্রশ্নের উত্তর দাও।

ক-বিভাগ (মান ঃ ২ × ১০ = ২০)

১। রেজোলুশন কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

২। Sector ফাংশনের ফরম্যাটসহ সংজ্ঞা লেখ।

উত্তর সংকেত ঃ অনুশীলনী এর অতি সংক্ষিপ্ত নং প্রশ্নোত্তর দ্রষ্টব্য।

৩। সফটওয়্যার ইন্টারাপ্ট কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ২ এর অতি সংক্ষিপ্ত ৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

৪। কপি কন্সট্রাক্টরের ফরম্যাটসহ সংজ্ঞা লেখ।

উত্তর সংকেত ঃ অনুশীলনী এর অতি সংক্ষিপ্ত নং প্রশ্নোত্তর দ্রষ্টব্য।

৫। ক্লাসের অ্যাক্সেসিবিলিটি মোডগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী এর অতি সংক্ষিপ্ত নং প্রশ্নোত্তর দ্রষ্টব্য।

৬। পাঁচটি জাভা 'কী'-ওয়ার্ডের নাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৫ এর অতি সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

৭। স্টেটমেন্ট বলতে কী বোঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৬ এর অতি সংক্ষিপ্ত ১০ নং প্রশ্নোত্তর দ্রষ্টব্য।

৮। মেথড ওভারলোডিং বলতে কী বোঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৭ এর অতি সংক্ষিপ্ত ৯ নং প্রশ্নোত্তর দ্রষ্টব্য।

৯। ইউজার ডিফাইন প্যাকেজ কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর অতি সংক্ষিপ্ত ১০ নং প্রশ্নোত্তর দ্রষ্টব্য।

১০। থ্রেড ক্লাস তৈরির প্রক্রিয়াগুলো কী কী?

উত্তর সংকেত ঃ অনুশীলনী ১০ এর অতি সংক্ষিপ্ত ৮ নং প্রশ্নোত্তর দ্রষ্টব্য।

খ-বিভাগ (মান ঃ ৩ × ১০ = ৩০)

১১। ট্রাডিশনাল প্রোগ্রামিং ল্যাঙ্গুয়েজের অসুবিধাগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১২। ফাংশন ওভারলোডিং-এর নিয়ম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর সংক্ষিপ্ত ১৭ নং প্রশ্নোত্তর দ্রষ্টব্য।

খ-বিভাগ (মান ঃ ৩ × ১০ = ৩০)

১১। জাভা প্রোগ্রামের বৈশিষ্ট্যগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৫ এর সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১২। গার্বেজ কালেকশন কী, বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর সংক্ষিপ্ত ১০ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৩। মাউস প্রোগ্রামিং বলতে কী বুঝায়? এটির বিবেচ্য বিষয়গুলো কী কী?

উত্তর সংকেত ঃ অনুশীলনী ২ এর অতি সংক্ষিপ্ত ৫ ও ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৪। ইন-লাইন ফাংশন বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ২১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৫। অপারেটর ওভারলোডিং কী? এটির সাধারণ নিয়মগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৩০ এবং সংক্ষিপ্ত ১৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৬। সি++ প্রোগ্রামের ইনহেরিটেন্সগুলোর ফরম্যাট লেখ।

উত্তর সংকেত ঃ ৪.৮ নং অনুচ্ছেদ (Inhreritance mode) দ্রষ্টব্য।

১৭। কনসোল I/O কী? ফরম্যাটেড অপারেশনের প্রয়োজনীয়তা কী?

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৫ ও ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৮। Friend Function কী? এটির ফরম্যাট লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর সংক্ষিপ্ত ২১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৯। Values ( ) এবং Values of ( ) Method দুটির পার্থক্য লেখ।

উত্তর সংকেত ঃ অনুশীলনী ১১ এর সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

২০। তিনটি সংখ্যার মাঝে ছোট সংখ্যা বের করার জন্য C++ এ Program লেখ।

উত্তর সংকেত ঃ নিজে চেষ্টা কর।

গ-বিভাগ (মান ঃ ৬ × ৫ = ৩০)

২১। কপি কনস্ট্রাক্টর ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর রচনামূলক ১২ নং প্রশ্নোত্তর দ্রষ্টব্য।

২২। বাংলাদেশের জাতীয় পতাকা অঙ্কনের একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ১ এর রচনামূলক ২ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৩। মেথড ওভারলোডিং ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর রচনামূলক ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৪। মাল্টিলেভেল ইনহেরিটেন্স ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর রচনামূলক ১৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৫। ক্লাশ ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর রচনামূলক ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৬। কনস্ট্রাক্টর ও ডিস্ট্রাক্টর ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর রচনামূলক ১১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৩। উদাহরণসহ একটি পূর্ণাঙ্গ ক্লাসের গঠন দেখাও।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর সংক্ষিপ্ত ২ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৪। ম্যাসেজ পাসিং প্রক্রিয়া চিত্রসহ বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর সংক্ষিপ্ত ৯ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৫। অপারেটর ওভারলোডিং-এর উদাহরণসহ বর্ণনা দাও।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৩০ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৬। Inline ফাংশন বলতে কী বুঝায়? উদাহরণসহ লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ২১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৭। 1, 2, 3, 5, 8, 13, 21 সিরিজটি নির্ণয়ের জন্য C++ ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ নিজে চেষ্টা কর।

১৮। জাভা ভাষার বৈশিষ্ট্যগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৫ এর সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৯। 1 + 3 + 5 + 7 + ... + 99 সিরিজটির মান নির্ণয়ের প্রোগ্রাম C++ ভাষায় রচনা কর।

উত্তর সংকেত ঃ নিজে চেষ্টা কর।

২০। ইনহেরিটেন্স কাকে বলে? এটির প্রকারভেদ লেখ।

উত্তর সংকেত ঃ ৪.৮ নং অনুচ্ছেদ দ্রষ্টব্য।

গ-বিভাগ (মান ঃ ৬ × ৫ = ৩০)

২১। মাল্টিলেভেল ইনহেরিটেন্স ব্যবহার করে C++ ভাষায় প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর রচনামূলক ১৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

২২। কপি কনস্ট্রাক্টর ব্যবহার করে Java ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ব্যবহারিক (জাভা প্রোগ্রামসমূহ) প্রোগ্রাম-১২ দ্রষ্টব্য।

২৩। 1 – 100 পর্যন্ত মৌলিক সংখ্যাগুলো নির্ণয়ের জন্য C++ ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ব্যবহারিক C++ (Pragram-11) দ্রষ্টব্য।

২৪। সুপার-ক্লাস ও সাব-ক্লাস ব্যবহার করে C++ ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর রচনামূলক ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৫। কী-বোর্ড থেকে একটি সংখ্যা নিয়ে তার গ্রেড নির্ণয় করার জন্য C ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুরূপ ব্যবহারিক (C প্রোগ্রামসমূহ) প্রোগ্রাম-১৫ দ্রষ্টব্য।

২৬। সফটওয়্যার বিবর্তনের ধাপগুলো বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর রচনামূলক ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৩। উদাহরণসহ একটি পূর্ণাঙ্গ ক্লাসের গঠন দেখাও।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর সংক্ষিপ্ত ২ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৪। ম্যাসেজ পাসিং প্রক্রিয়া চিত্রসহ বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর সংক্ষিপ্ত ৯ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৫। অপারেটর ওভারলোডিং-এর উদাহরণসহ বর্ণনা দাও।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৩০ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৬। Inline ফাংশন বলতে কী বুঝায়? উদাহরণসহ লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ২১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৭। 1, 2, 3, 5, 8, 13, 21 সিরিজটি নির্ণয়ের জন্য C++ ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ নিজে চেষ্টা কর।

১৮। জাভা ভাষার বৈশিষ্ট্যগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৫ এর সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৯। 1 + 3 + 5 + 7 + ... + 99 সিরিজটির মান নির্ণয়ের প্রোগ্রাম C++ ভাষায় রচনা কর।

উত্তর সংকেত ঃ নিজে চেষ্টা কর।

২০। ইনহেরিটেন্স কাকে বলে? এটির প্রকারভেদ লেখ।

উত্তর সংকেত ঃ ৪.৮ নং অনুচ্ছেদ দ্রষ্টব্য।

গ-বিভাগ (মান ঃ ৬ × ৫ = ৩০)

২১। মাল্টিলেভেল ইনহেরিটেন্স ব্যবহার করে C++ ভাষায় প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর রচনামূলক ১৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

২২। কপি কনস্ট্রাক্টর ব্যবহার করে Java ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ব্যবহারিক (জাভা প্রোগ্রামসমূহ) প্রোগ্রাম-১২ দ্রষ্টব্য।

২৩। 1 – 100 পর্যন্ত মৌলিক সংখ্যাগুলো নির্ণয়ের জন্য C++ ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ ব্যবহারিক C++ (Pragram-11) দ্রষ্টব্য।

২৪। সুপার-ক্লাস ও সাব-ক্লাস ব্যবহার করে C++ ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর রচনামূলক ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৫। কী-বোর্ড থেকে একটি সংখ্যা নিয়ে তার গ্রেড নির্ণয় করার জন্য C ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুরূপ ব্যবহারিক (C প্রোগ্রামসমূহ) প্রোগ্রাম-১৫ দ্রষ্টব্য।

২৬। সফটওয়্যার বিবর্তনের ধাপগুলো বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর রচনামূলক ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

বাংলাদেশ কারিগরি শিক্ষা বোর্ড, ঢাকা

ডিপ্লোমা-ইন-ইঞ্জিনিয়ারিং

৪র্থ পর্ব পরিপূরক পরীক্ষা-২০১৩

টেকনোলজি ঃ কম্পিউটার, কম্পিউটার সাইয়েন্স অ্যান্ড টেকনোলজি ও ডাটা টেলিকমিউনিকেশন অ্যান্ড নেটওয়ার্কিং

(২০১০ প্রবিধান)

বিষয় ঃ প্রোগ্রামিং ল্যাংগুয়েজ-২

বিষয় কোড ঃ ৬৬৪২

সময় ঃ ৩ ঘণ্টা পূর্ণমান ঃ ৮০

ক ও খ-বিভাগের **সকল** প্রশ্নের এবং গ-বিভাগের যে-কোন ৫ **(পাঁচ)** টি প্রশ্নের উত্তর দাও।

ক-বিভাগ (মান ঃ ২ × ১০ = ২০)

১। অবজেক্ট অরিয়েন্টেড প্রোগ্রাম বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

২। ইনহেরিটেন্স কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৯ নং প্রশ্নোত্তর দ্রষ্টব্য।

৩। রেজোলুশন বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

৪। ক্লাশ মেম্বার অ্যাক্সেসিবিলিটি মোডগুলোর নাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

৫। কপি কনস্ট্রাক্টর বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৩২ নং প্রশ্নোত্তর দ্রষ্টব্য।

৬। Class বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৩ এর অতি সংক্ষিপ্ত ৮ নং প্রশ্নোত্তর দ্রষ্টব্য।

৭। প্যাকেজ বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর অতি সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

৮। মেথড কী? মেথডের উপাদানগুলোর নাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর অতি সংক্ষিপ্ত ৫ ও ৭ নং প্রশ্নোত্তর দ্রষ্টব্য।

৯। Initgraph ( ) ফাংশনের কাজ কী?

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১০। জাভা বিন বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর অতি সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

খ-বিভাগ (মান ঃ ৩ × ১০ = ৩০)

১১। জাভা প্রোগ্রামের বৈশিষ্ট্যগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৫ এর সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১২। গার্বেজ কালেকশন কী, বর্ণনা কর।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর সংক্ষিপ্ত ১০ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৩। মাউস প্রোগ্রামিং বলতে কী বুঝায়? এটির বিবেচ্য বিষয়গুলো কী কী?

উত্তর সংকেত ঃ অনুশীলনী ২ এর অতি সংক্ষিপ্ত ৫ ও ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৪। ইন-লাইন ফাংশন বলতে কী বুঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ২১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৫। অপারেটর ওভারলোডিং কী? এটির সাধারণ নিয়মগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৩০ এবং সংক্ষিপ্ত ১৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৬। সি++ প্রোগ্রামের ইনহেরিটেন্সগুলোর ফরম্যাট লেখ।

উত্তর সংকেত ঃ ৪.৮ নং অনুচ্ছেদ (Inhreritance mode) দ্রষ্টব্য।

১৭। কনসোল I/O কী? ফরম্যাটেড অপারেশনের প্রয়োজনীয়তা কী?

উত্তর সংকেত ঃ অনুশীলনী ৪ এর অতি সংক্ষিপ্ত ৫ ও ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৮। Friend Function কী? এটির ফরম্যাট লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর সংক্ষিপ্ত ২১ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৯। Values ( ) এবং Values of ( ) Method দুটির পার্থক্য লেখ।

উত্তর সংকেত ঃ অনুশীলনী ১১ এর সংক্ষিপ্ত ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

২০। তিনটি সংখ্যার মাঝে ছোট সংখ্যা বের করার জন্য C++ এ Program লেখ।

উত্তর সংকেত ঃ নিজে চেষ্টা কর।

গ-বিভাগ (মান ঃ ৬ × ৫ = ৩০)

২১। কপি কনস্ট্রাক্টর ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর রচনামূলক ১২ নং প্রশ্নোত্তর দ্রষ্টব্য।

২২। বাংলাদেশের জাতীয় পতাকা অঙ্কনের একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ১ এর রচনামূলক ২ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৩। মেথড ওভারলোডিং ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর রচনামূলক ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৪। মাল্টিলেভেল ইনহেরিটেন্স ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর রচনামূলক ১৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৫। ক্লাশ ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর রচনামূলক ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৬। কনস্ট্রাক্টর ও ডিস্ট্রাক্টর ব্যবহার করে একটি সি++ প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর রচনামূলক ১১ নং প্রশ্নোত্তর দ্রষ্টব্য।

বাংলাদেশ কারিগরি শিক্ষা বোর্ড, ঢাকা

ডিপ্লোমা-ইন-ইঞ্জিনিয়ারিং

৪র্থ পর্ব পরিপূরক পরীক্ষা-২০১৪ (পরীক্ষার তারিখ ঃ ২/৭/২০১৪)

টেকনোলজি ঃ কম্পিউটার, ডাটা টেলিকমিউনিকেশন অ্যান্ড নেটওয়ার্ক ও কম্পিউটার সায়েন্স (২০১০ প্রবিধান)

বিষয় ঃ প্রোগ্রামিং ল্যাংগুয়েজ-২

বিষয় কোড ঃ ৬৬৪২

সময় ঃ ৩ ঘণ্টা পূর্ণমান ঃ ৮০

ক ও খ-বিভাগের **সকল** প্রশ্নের এবং গ-বিভাগের যে-কোন ৫ (পাঁচ) টি প্রশ্নের উত্তর দাও।

ক-বিভাগ (মান ঃ ২ × ১০ = ২০)

১। রেজোলুশন কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ১ এর অতি সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

২। Sector ফাংশনের ফরম্যাটসহ সংজ্ঞা লেখ।

উত্তর সংকেত ঃ অনুশীলনী এর অতি সংক্ষিপ্ত নং প্রশ্নোত্তর দ্রষ্টব্য।

৩। সফটওয়্যার ইন্টারাপ্ট কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ২ এর অতি সংক্ষিপ্ত ৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

৪। কপি কন্সট্রাক্টরের ফরম্যাটসহ সংজ্ঞা লেখ।

উত্তর সংকেত ঃ অনুশীলনী এর অতি সংক্ষিপ্ত নং প্রশ্নোত্তর দ্রষ্টব্য।

৫। ক্লাসের অ্যাক্সেসিবিলিটি মোডগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী এর অতি সংক্ষিপ্ত নং প্রশ্নোত্তর দ্রষ্টব্য।

৬। পাঁচটি জাভা 'কী'-ওয়ার্ডের নাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৫ এর অতি সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

৭। স্টেটমেন্ট বলতে কী বোঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৬ এর অতি সংক্ষিপ্ত ১০ নং প্রশ্নোত্তর দ্রষ্টব্য।

৮। মেথড ওভারলোডিং বলতে কী বোঝায়?

উত্তর সংকেত ঃ অনুশীলনী ৭ এর অতি সংক্ষিপ্ত ৯ নং প্রশ্নোত্তর দ্রষ্টব্য।

৯। ইউজার ডিফাইন প্যাকেজ কাকে বলে?

উত্তর সংকেত ঃ অনুশীলনী ৯ এর অতি সংক্ষিপ্ত ১০ নং প্রশ্নোত্তর দ্রষ্টব্য।

১০। থ্রেড ক্লাস তৈরির প্রক্রিয়াগুলো কী কী?

উত্তর সংকেত ঃ অনুশীলনী ১০ এর অতি সংক্ষিপ্ত ৮ নং প্রশ্নোত্তর দ্রষ্টব্য।

খ-বিভাগ (মান ঃ ৩ × ১০ = ৩০)

১১। ট্রাডিশনাল প্রোগ্রামিং ল্যাঙ্গুয়েজের অসুবিধাগুলো লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৩ এর সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১২। ফাংশন ওভারলোডিং-এর নিয়ম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর সংক্ষিপ্ত ১৭ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৩। সি++ এ ফাংশন ব্যবহারের বিবেচ্য বিষয়গুলো কী কী?

উত্তর সংকেত ঃ অনুশীলনী ৪ এর সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৪। সিস্টেমের বর্তমান সময় জানার জন্য সি ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ নিজে চেষ্টা কর।

১৫। লেটারেল বলতে কী বোঝায়? উদাহরণসহ লেখ।

উত্তর সংকেত ঃ ৫.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

১৬। মাল্টিটাস্কিং এবং মাল্টিথ্রেড প্রোগ্রামিং বলতে কী বোঝায়?

উত্তর সংকেত ঃ অনুশীলনী ১০ এর অতি সংক্ষিপ্ত ৫ ও ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৭। অ্যানোটেশন কাকে বলে? অ্যানোটেশন তৈরির ফরম্যাট লেখ।

উত্তর সংকেত ঃ অনুশীলনী ১১ এর অতি সংক্ষিপ্ত ৭ ও ৮ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৮। মেথড ব্যবহারের সুবিধাগুলো কী কী?

উত্তর সংকেত ঃ অনুশীলনী ৭ এর সংক্ষিপ্ত ৬ নং প্রশ্নোত্তর দ্রষ্টব্য।

১৯। JVM ব্যবহার করা হয় কেন? ব্যাখ্যা দাও।

উত্তর সংকেত ঃ অনুশীলনী ৫ এর সংক্ষিপ্ত ৯ নং প্রশ্নোত্তর দ্রষ্টব্য।

২০। আন্তঃথ্রেড কমিউনিকেশন পদ্ধতি ব্যাখ্যা কর।

উত্তর সংকেত ঃ অনুশীলনী ১০ এর সংক্ষিপ্ত ৪ নং প্রশ্নোত্তর দ্রষ্টব্য।

গ-বিভাগ (মান ঃ ৬ × ৫ = ৩০)

২১। বাংলাদেশের জাতীয় পতাকা অঙ্কনের জন্য সি ভাষায় প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ১ এর রচনামূলক ২ নং প্রশ্নোত্তর দ্রষ্টব্য।

২২। সুপার-ক্লাস ও সাব-ক্লাস ব্যবহার করে সি++ একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৮ এর রচনামূলক ১ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৩। মেথড ওভারলোডিং ব্যবহার করে একটি জাভা প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৭ এর রচনামূলক ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৪। ইন্টারফেস প্রক্রিয়ার বৈশিষ্ট্যাবলি লেখ।

উত্তর সংকেত ঃ ৯.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

২৫। প্রথম ২৫টি মৌলিক সংখ্যা নির্ণয়ের একটি জাভা ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৬ এর রচনামূলক ৫ নং প্রশ্নোত্তর দ্রষ্টব্য।

২৬। মাল্টিলেভেল ইনহেরিটেন্স ব্যবহার করে জাভা ভাষায় একটি প্রোগ্রাম লেখ।

উত্তর সংকেত ঃ অনুশীলনী ৪ এর রচনামূলক ১৩ নং প্রশ্নোত্তর দ্রষ্টব্য।

BDeBooks.com

E-BOOK

www.BDeBooks.com
FB.com/BDeBooksCom
BDeBooks.Com@gmail.com