ऑब्जेक्ट ओरिएंटेड प्रोग्रामिंग

बहुचयनात्मक

प्रश्न 1. बहुत प्रकार के रूप लेने की योग्यता कहलाती है।

(अ) वंशानुक्रम (Inheritance)
(ब) बहुरूपता (Polymorphism)
(स) सदस्य फंक्शन (Member function)
(द) संपुटीकरण (Encapsulation)

उत्तर: (ब) बहुरूपता (Polymorphism)

प्रश्न 2. किसी ऑब्जेक्ट के गुणधर्मों को बाहर निकालने की प्रक्रिया कहलाती है।

(अ) बहुरूपता (Polymorphism)
(ब) वंशानुक्रम (Inheritance)
(स) अमूर्तता (Abstraction)
(द) डेटा आच्छादन (Data hiding)

उत्तर: (द) डेटा आच्छादन (Data hiding)

प्रश्न 3. C++ की कौन सी सुविधाएँ इसे शक्तिशाली बनाती हैं?

(अ) सरल कार्यान्वयन (Easy implementation)
(ब) पुराने कोड का पुन: उपयोग (Reusing old code)
(स) नया कोड लिखना (Writing new code)
(द) ये सभी (All these)

उत्तर: (द) ये सभी (All these)

प्रश्न 4. निम्नलिखित में से क्या C++ में उपलब्ध OOP की सुविधा नहीं है?

(अ) संपुटीकरण (Encapsulation)
(ब) अमूर्तता (Abstraction)
(स) बहुरूपता (Polymorphism)
(द) अपवाद (Exceptions)

उत्तर: (द) अपवाद (Exceptions)

प्रश्न 5. ऑब्जेक्ट-ओरिएंटेड कहलाने के लिए प्रोग्रामिंग भाषा में आवश्यक है।

(अ) संपुटीकरण (Encapsulation)
(ब) अमूर्तता (Abstraction)
(स) बहुरूपता (Polymorphism)
(द) ये सभी (All these)

उत्तर: (द) ये सभी (All these)

प्रश्न 6. कौनसा कथन असत्य है?

(अ) किसी विशिष्ट कार्य को करने हेतु कोड का एक हिस्सा फंक्शन कहलाता है।
(ब) फंक्शन की सहायता से बड़े और जटिल प्रोग्राम को छोटे और सरल टुकड़ों में बाँटा जा सकता है।
(स) उपलब्ध कोड का प्रयोग करते हुए सामान्य कार्यों को करने हेतु फंक्शन सहायक हैं।
(द) प्रोग्राम के फंक्शन को मात्र एक ही बार काम में लिया जा सकता है।

उत्तर: (द) प्रोग्राम के फंक्शन को मात्र एक ही बार काम में लिया जा सकता है।

प्रश्न 7. किसी क्लास में परिभाषित फंक्शन के लिए क्या नाम है?

(अ) सदस्य चर (Member variable)
(ब) सदस्य फंक्शन (Member function)
(स) क्लास फंक्शन (Class function)
(द) क्लासिक फंक्शन (Classic function)

उत्तर: (ब) सदस्य फंक्शन (Member function)

प्रश्न 8. OOPS की किस अवधारणा का अर्थ है-मात्र आवश्यक सूचना को ही बाहर दर्शाना।

(अ) संपुटीकरण (Encapsulation)
(ब) अमूर्तता (Abstraction)
(स) डेटा आच्छादन (Data hiding)
(द) डेटा बंधन (Data binding)

उत्तर: (स) डेटा आच्छादन (Data hiding)

प्रश्न 9. वास्तविक समय के उदाहरणों, जैसे वाहन, कर्मचारी, के बारे में सूचना संग्रह तथा उनको बेहतर ढंग से समझने के लिए कौन-सा दृष्टिकोण ज्यादा बेहतर है?

(अ) प्रक्रियात्मक दृष्टिकोण
(ब) ऑब्जेक्ट ओरिएंटेड दृष्टिकोण
(स) मोड्यूलर दृष्टिकोण
(द) इनमें से कोई नहीं

उत्तर: (ब) ऑब्जेक्ट ओरिएंटेड दृष्टिकोण

प्रश्न 10. ऑब्जेक्ट-ओरिएंटेड प्रोग्रामिंग के लाभ

(अ) कोड का पुन: उपयोग
(ब) डेटा का बेहतर उपयोग
(स) त्रुटि रहित
(द) ये सभी

उत्तर: (द) ये सभी

अतिलघु उत्तरीय प्रश्न

प्रश्न 1. प्रक्रियात्मक प्रोग्रामिंग में असंरचित प्रोग्रामिंग की अपेक्षा त्रुटियों की सम्भावना कम होती है

क्योंकि – प्रोग्राम का आकार ” ………… किया जा सकता है।

उत्तर- छोटा।

प्रश्न 2. OOP में प्रथम O का तात्पर्य है …….. ।

उत्तर- ऑब्जेक्ट।

प्रश्न 3. वंशानुक्रम में डेटा और …. ” को नयी क्लास में लिया जा सकता है।

उत्तर- मेथड।

प्रश्न 4. C++ में बस एक ………….” होता है।

उत्तर- प्रयोक्ता परिभाषित डेटा प्रकार।

प्रश्न 5. पुनः प्रयोज्यता का लाभ है…….

उत्तर- पहले लिखे गए कोड का उपयोग कर पाना।

लघु उत्तरीय प्रश्न

प्रश्न 1. क्लास और ऑब्जेक्ट में अंतर बताइए।

उत्तर- क्लास (Class) Class एक user defined data type है। Class में variable के साथ-साथ उनसे related functions भी create कर सकते हैं। Class को परिभाषित करने के पश्चात् हम आवश्यकतानुसार उसके ऑब्जेक्ट वेरियेबल बना सकते हैं। ऑब्जेक्ट (Object)

Class type के variables को ऑब्जेक्ट (object) कहा जाता है। ऑब्जेक्ट, प्रोग्राम के निष्पादन के समय काम आता है। इसकी परिभाषा क्लास (class) के माध्यम से दी जाती है। ऑब्जेक्ट (object) के द्वारा class के वेरियेबल (variables) और फंक्शन्स (functions) को access करते हैं।

प्रश्न 2. प्रोसीजर और मेथड में अंतर कीजिए।

उत्तर- प्रोसीजर (Procedure) – प्रक्रियात्मक प्रोग्रामिंग (procedural programming) में एक मुख्य प्रोग्राम में अन्य प्रोग्रामों का उपयोग किया जा सकता है। इन प्रोग्रामों को सहायक प्रक्रियाओं (procedure) के रूप में समझा जा सकता है, जिन्हें हम प्रोसिजर कहते हैं।

मुख्य प्रोग्राम में जब किसी प्रोसिजर को कॉल किया जाता है तो प्रोग्राम का नियन्त्रण उस प्रोसिजर पर जाता है, कम्प्यूटर उस निर्देश-समूह को संचलित करता है और समाप्ति के पश्चात् नियंत्रण को मुख्य प्रोग्राम में वापिस जिस अनुदेश से भेजा था, उसके ठीक अगले अनुदेश पर भेज देता है।

प्रोसीजर की सुविधा होने से हम प्रोग्राम का आकार कम कर पाते हैं, साथ ही उसे त्रुटि-रहित भी कर पाते हैं।

मेथड (Method) – जब एक ऑब्जेक्ट कोई सन्देश प्राप्त करता है तब उस सन्देश के साथ वाला प्रोग्राम कोड निष्पादित होता है।

दूसरे शब्दों में कहें तो ये सन्देश निर्धारित करते हैं कि ऑब्जेक्ट का व्यवहार किस प्रकार का रहेगा और ऑब्जेक्ट में लिखा गया प्रोग्राम कोड निर्धारित करता है कि ऑब्जेक्ट क्या करेगा। सन्देश के साथ जुड़े हुए कोड को मेथड (method) कहते हैं।

जब ऑब्जेक्ट को कोई सन्देश प्राप्त होता है तब उस सन्देश में मेथड का नाम भी होता है, इस से निर्धारित होता है कि अब कौन-सा प्रोग्राम कोड निष्पादित होगा और तत्पश्चात निष्पादन हेतु नियंत्रण उस कोड को दे दिया जाता है। वह कोड निष्पादित होता है और उसका जो परिणाम है वह आगे भेज दिया जाता है।

प्रक्रियात्मक भाषाओं के सन्दर्भ में जिन्हें प्रोसीजर कहा जाता है, मेथड उसी प्रकार का निर्देशों का समूह है। मेथड का नाम, प्रोसीजर का नाम और मेथड में लिखा गया प्रोग्राम कोड, प्रोसीजर में लिखा गया कोड है। एक ऑब्जेक्ट को सन्देश भेजना, एक प्रोसीजर को काम में लेने जैसा ही है।

प्रश्न 3. C++ में टोकन क्या होते हैं?

उत्तर- टोकन (Tokens) – टोकन अक्षरों का समूह होता है। प्रोग्रामर इनका उपयोग करते हुए प्रोग्राम लिखता है। C++ भाषा में उपलब्ध टोकन हैं-Keywords] Identifies] Literal] Integer Constants.

कुँजी शब्द (Keywords) : C – भाषा में कुछ शब्द हैं जिन्हें पहले से ही भाषा में परिभाषित किया गया है।

ये सुरक्षित शब्द हैं जिनका अर्थ पहले से ही कम्पाइलर को ज्ञात है।

अभिज्ञानक (Identifiers) : C++ में विभिन्न डेटा तत्त्वों (data elements) के लिए प्रतीकात्मक नामों (Symbolic Name) का उपयोग करने की सुविधा प्रोग्रामर को उपलब्ध है। प्रतीकात्मक नाम को सामान्यतया अभिज्ञानक कहा जाता है।

इंटीजर कांस्टेंट (Integer constant) – इंटीजर कांस्टेंट अर्थात् पूर्ण संख्याएँ जिनमें कोई भिन्नात्मक भाग ना हो। C++ तीन प्रकार के इंटीजर कांस्टेंट की व्यवस्था प्रदान करता है।

प्रश्न 4. C++ में कैरेक्टर कांस्टेंट क्या होते हैं?

उत्तर- कैरेक्टर कांस्टेंट-एक अथवा अधिक कैरैक्टर जो कि एकल उद्धरण चिह्न में लिखे गए हों, जैसे '1', '4' इत्यादि। जिन कैरेक्टर्स को कीबोर्ड द्वारा सीधा प्रिंट नहीं किया जा सकता, जैसे टैब, बेक स्पेस इत्यादि, C++ में इस प्रकार के कैरेक्टर्स, एस्केप सीक्वेंस की सहायता से प्रदर्शित किये जा सकते हैं।

प्रश्न 5. डेटा प्रकार परिवर्तक (data type modifiers) क्या हैं?

उत्तर- डेटा प्रकार परिवर्तक, प्राइमरी डेटा टाईप (Primary Data Type) को क्वालीफाई (qualify) करते हैं। एक परिवर्तक (modifier) किसी बेस टाईप डेटा को बदलने की सुविधा देता है ताकि वह विभिन्न परिस्थितियों में fit हो सके।

निम्नांकित डेटा-प्रकार परिवर्तकों (data type modifiers) के उपयोग से कई मूल डेटा प्रकारों में परिवर्तन किया जा सकता है।

" Signed
" Unsigned
" Short
" Long

निबंधात्मक प्रश्न

प्रश्न 1. OOP की विशेषताओं का वर्णन कीजिए।

उत्तर- आब्जेक्ट ओरियेन्टेड प्रोग्रामिंग की विशेषताएँ :

पुनः प्रयोज्यता (Re usability) – किसी समस्या के हल के लिए लिखा गया प्रोग्राम, मात्र उसी समस्या का हल हो सकता है।

इसका अर्थ यह हुआ कि मिलती-जुलती समस्याओं के लिए भी हमें नए प्रोग्राम लिखने पड़ेंगे। ऐसा इसलिए होता है क्योंकि परम्परागत प्रोग्रामिंग में डेटा और प्रोग्राम अलग-अलग होते हैं। नयी समस्या के लिए नए प्रकार के डेटा भी हो सकते हैं,

और कुछ थोड़ा-बहुत बदलाव समस्या के आधार पर तर्कों में भी हो सकता है। परन्तु पुराना प्रोग्राम इसके लिए काम नहीं आ सकता।

ऑब्जेक्ट ओरिएंटेड में यह व्यवस्था है कि उपलब्ध समाधानों में बदलाव करते हुए नयी समस्याओं के लिए समाधान लिखे जा सकते हैं, अर्थात् नयी समस्या के लिए प्रोग्राम करने के लिए हमें प्रारम्भ से पुनः प्रयास करने की आवश्यकता नहीं है, हम पहले से बने हुए प्रोग्राम में आवश्यकतानुसार परिवर्तन करके वही समाधान प्राप्त कर सकते हैं।

पुनः प्रयोज्यता ऑब्जेक्ट ओरिएंटेड प्रोग्रामिंग की संभवतः महत्त्वपूर्ण विशेषताओं में से एक है।

एनकैप्सुलेशन (encapsulation) एवं एबस्ट्रेक्शन (abstraction) – प्रक्रियात्मक भाषाओं में डेटा अलग से परिभाषित किया जाता है और उससे काम लेने वाला फंक्शन अलग से। परन्तु ऑब्जेक्ट-ओरिएंटेड में, जैसा कि हमने जाना क्लास को परिभाषित करते समय हम डेटा और फंक्शन को एक साथ लिखते हैं।

इसे सम्पुटीकरण (encapsulation) कहा जाता हैं। एबस्ट्रेक्शन (abstraction) का तात्पर्य है कि एक क्लास में जो भी डेटा परिभाषित किया गया है, उसका उपयोग बिना अन्य विवरण और स्पष्टीकरण के उपयोग में लिया जा सकता है।

एनकैप्सुलेशन और एबस्ट्रेक्शन का लाभ ये होता है कि डेटा संरचना और संकारकों का उपयोग वैसा ही हो जाए जैसा प्रोग्रामर ने इन्हें परिभाषित करते समय सोचा है। इस प्रकार हम क्लास में दी गई सूचनाओं को छिपा (information hiding) सकते हैं जिस से प्रोग्रामिंग अधिक सुरक्षित हो जाती है।

बहुरूपता (Polymorphism) – किसी ऑब्जेक्ट का व्यवहार और कार्यान्वयन अलग-अलग होते हैं। एक ही सन्देश के लिए बहुत से ऑब्जेक्ट काम कर सकते हैं अथवा एक ही ऑब्जेक्ट विभिन्न रूपों में काम में लिया जा सकता है।

प्रोग्राम के निष्पादन के समय जैसी आवश्यकता होती है ऑब्जेक्ट का रूप वैसा हो सकता है। उदाहरण के लिए, addition एक क्लास है जिसे हम गणित की योग क्रिया (addition operation) के लिए परिभाषित कर रहे हैं।

इसमें add नामक एक मेथड है जिस में हम दो इन्टीजर वेरियेबल को जोड़ने का कोड लिखते हैं, इसी क्लास में add नाम से एक और मेथड है जिस में दो फ्लोट वेरियेबल को जोड़ने का कोड लिखते हैं। अब यदि प्रोग्राम के निष्पादन के समय add मैथड में दो इन्टीजर वेरियेबल का उपयोग करते हैं तो पहला मैथड

काम आयोग, और यदि दो फ्लोटवेरियेबल काम में लेते हैं तो दूसरा मैथड काम आएगा। यहाँ ध्यान देने वाली बात यही है कि add नाम से दो मैथड होते हुए भी निष्पादन के समय भेजे गए डेटा के आधार पर वांछित मैथड काम आएगा।

अर्थात् एक ही नाम अलग-अलग रूप में काम आ रहा है। बहुरूपता की व्यवस्था के कारण, प्रेषक ऑब्जेक्ट और प्रापक ऑब्जेक्ट के मध्य संदेशों के आदान-प्रदान संभव हो जाता है।

वंशानुक्रम (Inheritance) – वंशानुक्रम (inheritance), ऑब्जेक्ट-ओरिएंटेड प्रोग्रामिंग की एक और महत्त्वपूर्ण संकल्पना है। एक क्लास को परिभाषित करने के पश्चात् यदि हमें वैसी ही एक और क्लास को और परिभाषित करना हो एक ऐसी क्लास जिसके गुणधर्म वही हों जो पहली क्लास के हैं और साथ में कुछ और गुणधर्म भी हम जोड़ना चाहें-वंशानुक्रम की व्यवस्था से हम ऐसा कर पाएँगे।

ऐसी व्यवस्था में हम पुनः प्रयोज्यता का भी उपयोग कर रहे हैं, इस कारण से दुबारा से वही सारा काम करने की आवश्यकता नहीं है जो पहली क्लास को परिभाषित करते समय किया था। इससे समय भी बचता है और प्रोग्राम के रख रखाव में भी सुविधा होती है।

किसी क्लास के गुणधर्म लेने वाली क्लास को सब क्लास (sub class) या डिराइव्ड क्लास (derived class) कहा जाता है और जिस क्लास से गुणधर्म आगे लिए जाते हैं उसे सुपर क्लास (superclass) या बेस क्लास (base class) कहा जाता है।

प्रश्न 2. C++ के मूल डेटा प्रकार समझाइए।

उत्तर- C++ में मूल डेटा प्रकार (Basic Data Type)
Booleao : bool
Character : char
Integer : int
Floating point : float
Double floating point : double
Valueless : void
Wide character : wchar_t

निम्नांकित डेटा-प्रकार परिवर्तकों (data type modifiers) के उपयोग से कई मूल डेटा प्रकारों में परिवर्तन किया जा सकता है :

.. signed
.. unsigned
.. short
.. long

डेटा का प्रकार और कम्प्यूटर की मैमोरी में डेटा संग्रहीत करने के लिए कितनी मैमोरी की आवश्यकता होगी, तथा न्यूनतम व अधिकतम मान क्या. हो सकते हैं, इसे निम्नांकित तालिका में दर्शाया गया है :

Type	Typical bit width	Typical range
char	1byte	−128+127
unsigned char	1byte	0to255
int	4bytes	−2147483648to2147483647
signed int	4bytes	−2147483648to2147483647
short int	2bytes	−32768to32767
unsigned short int	Range	0to65.535
long int	4bytes	−2.147,483,648to2,147,483,647
unsigned long int	4bytes	0to4,294,967,295
float	4bytes	+/-3-4e+/-38(~7digits)
double	8bytes	+/-1-7e+/-308(~15digits)
long double	8bytes	+/-1-7e+/-308(~15digits)
wchar_t	2 or 4 bytes	1wide character

प्रश्न 3. cin एवं cout के प्रयोग को उदाहरण की सहायता से समझाइए।

उत्तर- cin ऑब्जेक्ट-कीबोर्ड के माध्यम से प्रयोक्ता द्वारा एक मान को इनपुट करने के लिए cin का उपयोग किया जा सकता है। कीबोर्ड द्वारा प्राप्त मान को मेमोरी में संग्रहीत करने के लिए > निष्कर्षण (extraction) संकारक का उपयोग किया जाना आवश्यक है।

cin>>marks;// कीबोर्ड से डेटा प्राप्त होगा वह marks चर में संग्रहीत हो जाएगा।

cout ऑब्जेक्ट
cout का उपयोग सन्देश को स्क्रीन पर प्रदर्शित करने के लिए << संकारक (operator) के संयोजन के साथ किया जाता है।

cout<<"Hello world";// स्क्रीन पर Hello world प्रदर्शित करता है
cout<<250;//250 की संख्या स्क्रीन पर प्रदर्शित करता है।
cout<<sum;// वेरिएबल sum के मान को स्क्रीन पर प्रदर्शित करता है।

यदि वेरियेबल और कांस्टेंट के संयोजन को साथ में प्रदर्शित करना हो तो << का उपयोग एक से अधिक बार किया जा सकता है।

cout<< Area of Crop Field is "<<area<<"'square meter";

प्रश्न 4. Encapsulation एवं Abstraction में अंतर स्पष्ट कीजिए।

उत्तर- एनकैप्सुलेशन (encapsulation) – प्रक्रियात्मक भाषाओं में डेटा अलग से परिभाषित किया जाता है और उसे काम लेने वाले फंक्शन अलग से। परन्तु, ऑब्जेक्ट-ओरिएंटेड प्रोग्रामिंग में, जैसा कि हमने जाना क्लास को परिभाषित करते समय हम डेटा और फंक्शन को एक साथ लिखते हैं।

इसे सम्पुटीकरण (encapsulation) कहा जाता है। डेटा व डेटा पर perform होने वाले operations के functions को एक यूनिट के रूप में bind करके ऑब्जेक्ट की क्लास (class) बनाने की प्रक्रिया ही encapsulation है।

इस आधार पर बनने वाली क्लास (class) के डेटा को केवल उसी क्लास (class) में define किये गए member function ही access कर सकते हैं। इन member function के अलावा कोई भी external function उस specific class के data को access नहीं कर सकता।

एबस्ट्रेक्शन (Abstraction) – एबस्ट्रेक्शन (Abstraction) का तात्पर्य है कि एक क्लास में जो भी डेटा परिभाषित किया गया है, उसका उपयोग बिना अन्य विवरण और स्पष्टीकरण के उपयोग में लिया जा सकता है।

एनकैप्सुलेशन और एबस्ट्रेक्शन का लाभ यह होता है कि डेटा संरचना और संकारकों का उपयोग वैसा ही हो जाए जैसा प्रोग्रामर ने इन्हें परिभाषित करते समय सोचा है। इस प्रकार हम क्लास में दी गई सूचनाओं को छिपा (information hiding) सकते हैं जिससे प्रोग्रामिंग अधिक सुरक्षित हो जाती है।

एबस्ट्रेक्शन (Abstraction) एक ऐसी प्रक्रिया होती है, जिसमें किसी समस्या से संबंधित जरूरी बातों को बिना जरूरी बातों से अलग किया जाता है। फिर उन जरूरी बातों को समस्या के किसी ऑब्जेक्ट (object) की properties के रूप में describe किया जाता है।

अन्य महत्वपूर्ण प्रश्न

अतिलघूत्तरात्मक प्रश्न

प्रश्न 1. प्रोग्राम से आप क्या समझते हैं?

उत्तर- किसी निश्चित कार्य को पूरा करने हेतु लिखे गए अनुदेशों (instructions) के समूह को प्रोग्राम कहते हैं।

प्रश्न 2. मुख्य प्रोग्राम का क्या अर्थ है?

उत्तर- मुख्य प्रोग्राम का अर्थ है-आदेशों का एक समूह है जिसमें डेटा पूरे प्रोग्राम के दौरान उपलब्ध रहता है और उसे बदला जा सकता है।

प्रश्न 3. प्रक्रियात्मक प्रोग्रामिंग को एक विशेषता बताइए।

उत्तर- प्रक्रियात्मक प्रोग्रामिंग की सहायता से प्रोग्राम का आकार छोटा किया जा सकता है।

प्रश्न 4. किसी प्रोग्रामिंग में प्रोसीजर का क्या महत्त्व है?

उत्तर- प्रोग्रामिंग में, प्रोसीजर की सुविधा होने से हम प्रोग्राम का आकार कम कर पाते हैं, साथ ही उसे त्रुटि-रहित भी कर पाते हैं।

प्रश्न 5. OOP के आने से पहले, किस भाषा का प्रयोग होता था?

उत्तर- OOP के आने से पहले, प्रक्रियात्मक भाषाओं का प्रयोग होता था।

प्रश्न् 6. ऑब्जेक्ट क्या होते हैं?

उत्तर- Class type के variables को ऑब्जेक्ट (object) कहा जाता है। एक ऑब्जेक्ट में आदेश और डेटा संग्रहित किये जा सकते हैं।

प्रश्न 7. प्रेषक ऑब्जेक्ट (sender object) क्या होते हैं?

उत्तर- डेटा अथवा आदेश के माध्यम से ऑब्जेक्ट क्रिया करने के लिए अनुरोध करने वाला ऑब्जेक्ट, प्रेषक ऑब्जेक्ट (sender object) कहलाता है।

प्रश्न 8. सम्पुटीकरण (encapsulation) किसे कहते हैं?

उत्तर- ऑब्जेक्ट-ओरिएंटेड प्रोग्रामिंग में, क्लास को परिभाषित करते समय हम डेटा और फंक्शन को एक साथ लिखते हैं।

इसे ही सम्पुटीकरण (encapsulation) कहते हैं।

प्रश्न 9. एनकैप्सुलेशन और एबस्ट्रेक्शन का क्या लाभ है?

उत्तर- एनकैप्सुलेशन और एबस्ट्रेक्शन का लाभ ये होता है कि डेटा संरचना और संकारकों का उपयोग वैसा ही हो जाए जैसा प्रोग्रामर ने इन्हें परिभाषित करते समय सोचा है।

प्रश्न 10. सब क्लास (sub class) या डिराइव्ड क्लास (derived class) क्या होती है?

उत्तर- किसी क्लास के गुणधर्म लेने वाली क्लास को सब क्लास (sub class) या डिराइब्ड क्लास (derived class) कहा जाता है।

प्रश्न 11. सुपर क्लास (super class) या बेस क्लास (base class) क्या होती है?

उत्तर- जिस क्लास से गुणधर्म आगे लिये जाते हैं, उसे सुपर क्लास (super class) या बेस क्लास (base class) कहा जाता है।

प्रश्न 12. क्लास को बनाने के लिए किस कीवर्ड का उपयोग किया जाता है?

उत्तर- क्लास को बनाने के लिए class कीवर्ड उपयोग किया जाता है।

प्रश्न 13. कुँजी शब्द (keywords) क्या होते हैं?

उत्तर- C++ भाषा में कुछ शब्द हैं जिन्हें पहले से ही भाषा में परिभाषित किया गया है। ये सुरक्षित शब्द हैं जिनका अर्थ पहले से ही कम्पाइलर को ज्ञात है।

प्रश्न 14. अचर (constant) क्या होते हैं?

उत्तर- वे डेटा तत्त्व जिनका मान पूरे प्रोग्राम में कभी बदला नहीं जा सकता, स्थिरांक अथवा अचर (constant) कहलाते हैं।

प्रश्न 15. स्ट्रिंग कांस्टेंट (string constant) की परिभाषा बताइए।

उत्तर- दोहरे उद्धरण चिह्न (double quotation mark) में लिखी गयी अक्षरों की श्रृंखला को स्ट्रिंग कांस्टेंट कहते हैं।

प्रश्न 16. हैडर फाइल iostream.h का क्या उपयोग है?

उत्तर- C++ की स्टैण्डर्ड लाइब्रेरी में एक हैडर फाइल से पढ़ने और स्क्रीन पर प्रदर्शि करने के लिए किया जाता है।

प्रश्न 17. OOP के मूल में क्या सोच है?

उत्तर- OOP के मूल में सोच यह है कि एक प्रोग्राम के प्रोसीजर, फंक्शन या उप-प्रोग्राम और उनमें काम आने वाले डेटा एक साथ होंगे तो उनको काम में लेना सरल हो जाएगा।

लघु उत्तरीय प्रश्न

प्रश्न 1. प्रक्रियात्मक प्रोग्रामिंग से आप क्या समझते हैं?

उत्तर- प्रक्रियात्मक प्रोग्रामिंग (Procedural Programming)-प्रक्रियात्मक प्रोग्रामिंग की सहायता से प्रोग्राम का आकार छोटा किया जा सकता है, उसमें त्रुटियों की सम्भावना कम होती है, और उसका रख-रखाव भी आसान हो जाता है। मुख्य प्रोग्राम में जब किसी प्रोसीजर को कॉल किया जाता है तो प्रोग्राम का

नियंत्रण उस प्रोसीजर पर जाता है, कम्प्यूटर उस निर्देश-समूह को संचालित करता है, और समाप्ति के पश्चात् नियंत्रण को मुख्य प्रोग्राम में वापिस जिस अनुदेश से भेजा था, उसके ठीक अगले अनुदेश पर भेज देता है।

प्रोग्राम का नियंत्रण, जिस अनुदेश के द्वारा प्रोसीजर के निष्पादन के लिए भेजा गया था, निष्पादन के पश्चात् उसके अगले अनुदेश पर आता है।

इस प्रकार की प्रोग्रामिंग में, प्रोसीजर की सुविधा होने से हम प्रोग्राम का आकार कम कर पाते हैं, साथ ही उसे त्रुटि-रहित भी कर पाते हैं।

प्रश्न 2. प्रक्रियात्मक भाषाओं के प्रमुख लाभ बताइए।

उत्तर- प्रक्रियात्मक भाषाओं के लाभ

- सामान्य प्रोग्रामिंग के लिए बहुत अच्छी होती है।
- बहुत सी समस्याओं के लिए कोड उपलब्ध हो सकता है, तो हो सकता है कि पुनः प्रोग्रामिंग (reprogramming) की आवश्यकता न पड़े।
- सीपीयू पर प्रोग्राम चलाना हो उसकी विशेष जानकरी होना आवश्यक नहीं है-जैसा कि मशीन भाषा में आवश्यक होता है।
- दूसरे कम्पाइलर की सहायता से किसी अन्य सीपीयू पर इसी प्रोग्राम को चलाया जा सकता है।

प्रश्न 3. प्रक्रियात्मक भाषाओं की सीमाएं बताइए।

उत्तर- प्रक्रियात्मक भाषाओं की सीमाएं

- बहुत सारी भाषाओं की उपलब्धता होने से प्रोग्रामर को किसी एक भाषा में बंध कर रहना होता है, और कोबोल के लिए अलग-अलग प्रकार की विशेषज्ञता की आवश्यकता होती है।
- भाषा-विशेष के प्रोग्रामिंग अनुदेशों की सूक्ष्म जानकारी और ज्ञान आवश्यक होता है।
- कृत्रिम बुद्धिमत्ता (artificial intelligence) की आवश्यकता वाली समस्याओं के लिए जहाँ तर्क अस्पष्ट हों (fuzzy logic) प्रक्रियात्मक भाषाएँ अनुकूल नहीं हैं।

प्रश्न 4. ऑब्जेक्ट ओरिएंटेड प्रोग्रामिंग के लाभ बताइए।

उत्तर- ऑब्जेक्ट-ओरिएंटेड प्रोग्रामिंग के लाभ

- प्रोग्रामिंग करना आसान है। एक क्लास को पूरी तरह से जाँचने के बाद काम में लेने की सुविधा के कारण त्रुटि होने की सम्भावना कम हो जाती है।
- क्लास को एक "ब्लैक बॉक्स" के रूप में माना जा सकता है, उसकी आतंरिक संसाधन व्यवस्था की जानकारी के बिना, उसमें उपलब्ध मेथड्स को उपयोग किया जा सकता है।
- एक ही प्रकार के प्रोग्राम को लिखने एवं जाँचने का अनावश्यक प्रयास बच जाता है।
- अपनी आवश्यकता के अनुरूप क्लास को सीधे काम में लिया जा सकता है। दूसरे शब्दों में कहें तो प्रोग्राम कोड का पुन: उपयोग किया जा सकता है।
- कोड को किसी अन्य कम्पाइलर के माध्यम से दूसरे सीपीयू के अनुरूप बनाया जा सकता है अर्थात् कोड पोर्टेबिलिटी की सुविधा मिलती है।

प्रश्न 5. परिज्ञापक क्या होते हैं?

उत्तर- परिज्ञापक (Identifiers)-C++ में विभिन्न डेटा तत्त्वों (data elements) के लिए प्रतीकात्मक नामों (symbolic name) का उपयोग करने की सुविधा प्रोग्रामर को उपलब्ध है।

प्रतीकात्मक नाम को सामान्यतया परिज्ञापक कहा जाता है।

C++ के अक्षर समुच्चय से अक्षर लेकर ये नाम बनाए जाते हैं। नाम बनाने के लिए नियम इस प्रकार हैं

- एक परिज्ञापक के नाम में अल्फाबेट्स (A-Z, a-z), अंक (0–9), और/अथवा अंडर स्कोर (_) हो सकता है।
- प्रथमाक्षर संख्या नहीं हो सकता।
- यह सुरक्षित शब्द नहीं होना चाहिए।

प्रश्न 6. C++ में कितने प्रकार के इंटीजर कांस्टेंट की व्यवस्था होती है?

उत्तर- इंटीजर कांस्टेंट-इंटीजर कांस्टेंट अर्थात् पूर्ण संख्याएँ जिनमें कोई भिन्नात्मक भाग ना हो। C++ तीन प्रकार के इंटीजर कांस्टेंट की व्यवस्था प्रदान करता है।

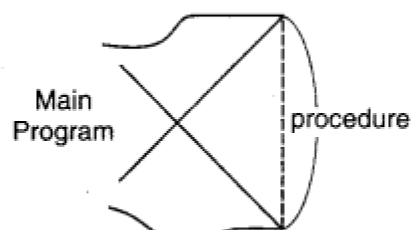
- डेसीमल इंटीजर कांस्टेंट : संख्याएँ, प्रथम अंक (0) शून्य नहीं हो सकता। जैसे 78,-168,+4
- ऑक्टल इंटीजर कांस्टेंट : संख्याएँ, प्रथम अंक (0) शून्य होता है। जैसे 014
- हेक्साडेसीमल इंटीजर कांस्टेंट : संख्याएँ, प्रथम दो अक्षर (0x) अथवा (0X) होते हैं। जैसे 0X24

निबंधात्मक प्रश्न

प्रश्न 1. प्रक्रियात्मक प्रोग्रामिंग को चित्र सहित समझाइए।

उत्तर- प्रक्रियात्मक प्रोग्रामिंग (Procedural programming) – प्रक्रियात्मक प्रोग्रामिंग की सहायता से प्रोग्राम का आकार छोटा किया जा सकता है, उसमें त्रुटियों की सम्भावना कम होती है, और उसका रख-रखाव भी आसान हो जाता है।

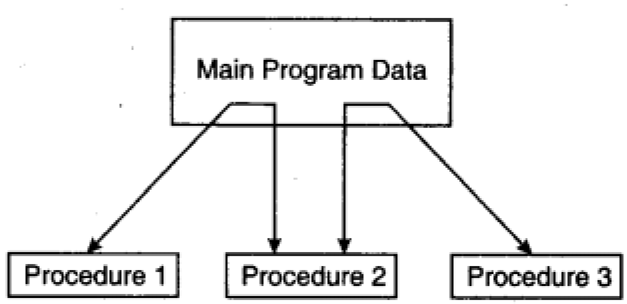
मुख्य प्रोग्राम में जब किसी प्रोसीजर को कॉल किया जाता है तो प्रोग्राम का नियंत्रण उस प्रोसीजर पर जाता है, कम्प्यूटर उस निर्देश-समूह को संचालित करता है, और समाप्ति के पश्चात् नियंत्रण को मुख्य प्रोग्राम में वापिस जिस अनुदेश से भेजा था, उसके ठीक अगले अनुदेश पर भेज देता है।



चित्र–3.1–प्रोसीजर का निष्पादन

प्रोग्राम का नियंत्रण, जिस अनुदेश के द्वारा प्रोसीजर के निष्पादन के लिए भेजा गया था, निष्पादन के पश्चात् उसके अगले अनुदेश पर आता है।

इस प्रकार की प्रोग्रामिंग में, प्रोसीजर की सुविधा होने से हम प्रोग्राम का आकार कम कर पाते हैं, साथ ही उसे त्रुटि-रहित भी कर पाते हैं।



चित्र–3.2–प्रक्रियात्मक प्रोग्रामिंग

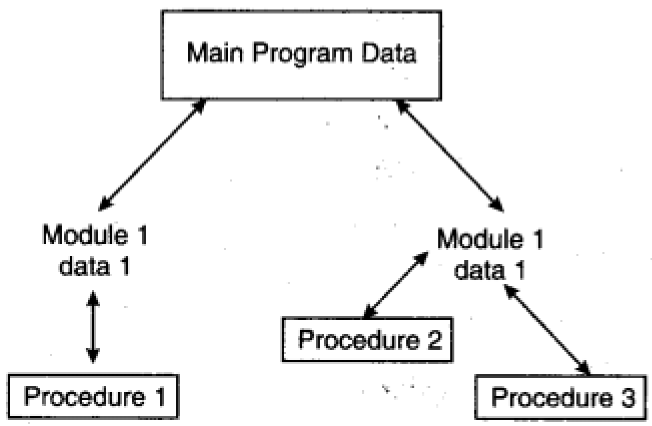
अब हम ऐसा भी मान सकते हैं कि एक प्रोग्राम बहुत से प्रोसीजर की श्रृंखला है। मुख्य प्रोग्राम से प्रत्येक प्रोसीजर को कॉल किया जाता है, डेटा को संसाधित किया जाता है और परिणाम मुख्य प्रोग्राम को प्रेषित कर दिए जाते हैं। इस प्रकार जब पूरी श्रृंखला सम्पूर्ण हो जाती है तब मुख्य प्रोग्राम से अंतिम परिणाम प्राप्त हो जाता है।मुख्य प्रोग्राम, प्रोसिजरों को डेटा प्राचलों के रूप में देता और लेता है।

मुख्य प्रोग्राम के नियंत्रण में-प्रोसिजरों का निष्पादन (execution) एवं डेटा का आदान-प्रदान होता है।

प्रश्न 2. मोड्यूलर प्रोग्रामिंग के विषय में चित्र सहित बताइए।

उत्तर- मोड्यूलर प्रोग्रामिंग-मोड्यूलर प्रोग्रामिंग में प्रोसिजरों को सामूहिक रूप से एक मोड्यूल के रूप संग्रहीत किया जाता है। अपनी सुविधा के अनुसार, एक मोड्यूल में समान प्रकार के संसाधन करने वाले प्रोसिजरों को रखा जाता है।

संग्रहीत प्रोसिजरों को मुख्य प्रोग्राम में सीधे काम लिया जा सकता है। मुख्य प्रोग्राम बहुत से छोटे टुकड़ों में बंट जाता है और प्रोसिजरों के माध्यम से डेटा संसाधन होता है।



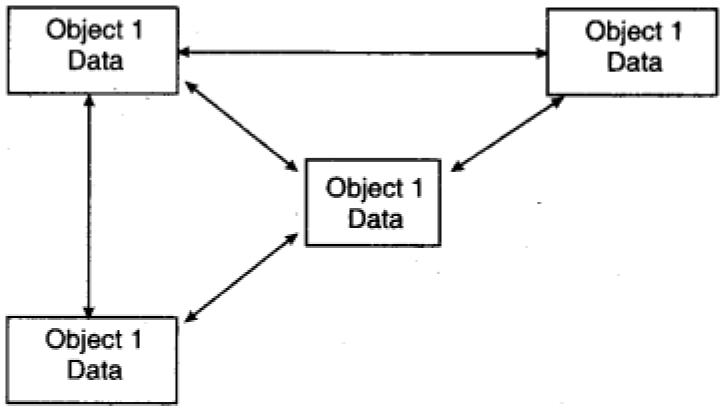
चित्र 3.3–मोड्यूलर प्रोग्रामिंग

प्रत्येक मोड्यूल का स्वयं का डेटा होता है। जब एक मोड्यूल के प्रोसिजरों का उपयोग किया जाता है तब प्रत्येक मोड्यूल अपनी आंतरिक डेटा व्यवस्था को बनाए रखता है। परन्तु एक मोड्यूल एक प्रोग्राम में एक समय में एक बार ही हो सकता है।

प्रश्न 3. ऑब्जेक्ट ओरिएंटेड प्रोग्रामिंग (OOP) के विषय में चित्र सहित समझाइए।

उत्तर- ऑब्जेक्ट ओरिएंटेड प्रोग्रामिंग (Object Oriented Programming OOP) ऑब्जेक्ट ओरिएंटेड प्रोग्रामिंग (Object Oriented Programming), प्रक्रियात्मक प्रोग्रामिंग (procedural programming) से भिन्न है। ऑब्जेक्ट पर आधारित भाषाओं में प्रोग्राम क्लास और ऑब्जेक्ट पर आधारित होते हैं। इनका उपयोग-प्रयोग हम करते हैं, इनमें लिखे मेथड्स के माध्यम से।

OOP के आने से पहले, प्रक्रियात्मक भाषाओं का प्रयोग होता था। इन भाषाओं में प्रोग्रामिंग अनुदेश और डेटा अलग-अलग होते हैं। इसी कारण, प्रोग्रामों के गलत होने की सम्भावना अधिक रहती है। यह समस्या और गंभीर हो जाती है जब प्रोग्राम की लम्बाई अधिक हो, अर्थात् उसमें अनुदेशों की संख्या ज्यादा हो,



चित्र 3.4–ऑब्जेक्ट ओरिएन्टेड प्रोग्रामिंग

अथवा जिस समस्या का समाधान करने हेतु प्रोग्राम लिखा गया है, वह समस्या स्वयं में ही जटिल हो। OOP के मूल में सोच यह है कि एक प्रोग्राम के प्रोसीजर, फंक्शन या उप-प्रोग्राम और उनमें काम आने वाले डेटा एक साथ होंगे तो उनको काम में लेना सरल हो जाएगा।

प्रश्न 4. समस्या समाधान का ऑब्जेक्ट ओरिएंटेड दृष्टिकोण समझाइए तथा ऑब्जेक्ट ओरिएंटेड प्रोग्रामिंग के लाभ बताइए।

उत्तर- समस्या समाधान का ऑब्जेक्ट ओरिएंटेड दृष्टिकोण (Object Oriented Problem Solving Approach) – अपने दैनिक जीवन में हम जिस प्रकार अपने कार्य करते हैं अथवा समस्याओं के समाधान ढूँढ़ते हैं, उसी प्रकार का दृष्टिकोण ही ऑब्जेक्ट-ओरिएंटेड प्रोग्रामिंग उपयोग आता है।

समस्या के समाधान में सहायक होने वाले ऑब्जेक्ट्स की पहचान करना और उन्हें निश्चित क्रम में उपयोग में लेना, अपनी समस्याओं के समाधान के लिए साधारणया हम ऐसा ही करते हैं। समस्या में आने वाले ऑब्जेक्ट्स के बारे में सोचते हैं और उनका उपयोग किस प्रकार करेंगे जिससे कि समस्या का समाधान हो सके।

दूसरे शब्दों में, ऑब्जेक्ट-ओरिएंटेड समस्या के समाधान में हम ऐसे ऑब्जेक्ट्स बनाते हैं जिनसे उस समस्या का समाधान हो सके।

ऑब्जेक्ट को भेजे गए सन्देश के आधार पर विभिन्न संक्रियाएँ होती हैं जिनसे समस्या का समाधान हो जाता है।

ऑब्जेक्ट-ओरिएंटेड समस्या के समाधान की प्रक्रिया को चार चरणों में विभाजित किया जा सकता है :

1. समस्या को पहचानना (Identify the problem)

2. समाधान हेतु आवश्यक ऑब्जेक्ट्स को पहचानना (Identify the objects needed for the solution)

3. ऑब्जेक्ट्स को भेजे जाने वाले संदेशों को पहचानना (Identify messages to be sent to the objects)

4. उन संदेशों के क्रम का निर्माण करना जिस से समस्या का समाधान प्राप्त हो (Create a sequence of messages to the objects that solve the problem)

ऑब्जेक्ट-ओरिएंटेड प्रोग्रामिंग के लाभ

1. प्रोग्रामिंग करना आसान है। एक क्लास को पूरी तरह से जाँचने के बाद काम में लेने की सुविधा के कारण त्रुटि होने की सम्भावना कम हो जाती है।

2. क्लास को एक “ब्लैक बॉक्स” के रूप में माना जा सकता है, उसकी आंतरिक संसाधन व्यवस्था की जानकारी के बिना, उसमें उपलब्ध मेथड्स को उपयोग किया जा सकता है।

3. एक ही प्रकार के प्रोग्राम को लिखने एवं जाँचने का अनावश्यक प्रयास बच जाता है। अपनी आवश्यकता के अनुरूप _क्लास को सीधे काम में लिया जा सकता है। दूसरे शब्दों में कहें तो प्रोग्राम कोड का पुन: उपयोग किया जा सकता है।

4. कोड को किसी अन्य कम्पाइलर के माध्यम से दूसरे सीपीयू के अनुरूप बनाया जा सकता है अर्थात् कोड पोर्टेबिलिटी की सुविधा मिलती है।