Biyani Girls College
I Internal Examination Solution Sept. 2019
Class: BCA II
Subject- Object Oriented Concepts (Using 'C++')

BCA- 206(A)
MM: 40 **Set: B**

[I] Very Short Answer Questions (Max 40 words). **(5*2=10)**

1. What is Overloading? Explain with the help of Example.

An overloaded declaration is a declaration that is declared with the same name as a previously declared declaration in the same scope, except that both declarations have different arguments and obviously different definition (implementation).

When you call an overloaded **function** or **operator**, the compiler determines the most appropriate definition to use, by comparing the argument types you have used to call the function or operator with the parameter types specified in the definitions.

2. Discuss the role of class in Object Oriented Programming?

In object-oriented programming, a **class** is an extensible program-code-template for creating objects, providing initial values for state (member variables) and implementations of behavior (member functions or methods). In many languages, the class name is used as the name for the class (the template itself), the name for the default constructor of the class (a subroutine that creates objects), and as the type of objects generated by instantiating the class; these distinct concepts are easily conflated.[2]

When an object is created by a constructor of the class, the resulting object is called an instance of the class, and the member variables specific to the object are called instance variables, to contrast with the class variables shared across the class.

3. What do you understand by structured programmming?

Structured programming is a programming paradigm aimed at improving the clarity, quality, and development time of a computer program by making extensive use of the structured control flow constructs of selection (if/then/else) and repetition (while and for), block structures, and subroutines.

4. What is typedef ? Give syntax for typedef.

The C++ programming language provides a keyword called **typedef**, which you can use to give a type a new name. Following is an example to define a term **BYTE** for one-byte numbers −

typedef unsigned char BYTE;

After this type definition, the identifier BYTE can be used as an abbreviation for the type **unsigned char, for example.**.

BYTE b1, b2;

By convention, uppercase letters are used for these definitions to remind the user that the type name is really a symbolic abbreviation, but you can use lowercase, as follows −

typedef unsigned char byte;

5. What is Polymorphism? Give an example.

The word polymorphism means having many forms. In simple words, we can define polymorphism as the ability of a message to be displayed in more than one form.
This is called polymorphism.
Polymorphism is considered as one of the important features of Object Oriented Programming.

In C++ polymorphism is mainly divided into two types:

- Compile time Polymorphism
- Runtime Polymorphism

[II] Short Answer Questions (Max 80 words). (2*5=10)

1. What are Operators and write briefly about various Operators available in C++?

An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations. C++ is rich in built-in operators and provide the following types of operators −

- Arithmetic Operators
- Relational Operators
- Logical Operators
- Bitwise Operators
- Assignment Operators
- Misc Operators

This chapter will examine the arithmetic, relational, logical, bitwise, assignment and other operators one by one.

Arithmetic Operators

There are following arithmetic operators supported by C++ language −

Assume variable A holds 10 and variable B holds 20, then −

Relational Operators

There are following relational operators supported by C++ language

Assume variable A holds 10 and variable B holds 20, then −

Logical Operators

There are following logical operators supported by C++ language.

Assume variable A holds 1 and variable B holds 0, then −

Bitwise Operators

Bitwise operator works on bits and perform bit-by-bit operation. The truth tables for &, |, and ^ are as follows −

Assume if A = 60; and B = 13; now in binary format they will be as follows −

A = 0011 1100

B = 0000 1101

A&B = 0000 1100

A|B = 0011 1101

A^B = 0011 0001

~A = 1100 0011

The Bitwise operators supported by C++ language are listed in the following table. Assume variable A holds 60 and variable B holds 13, then −

Assignment Operators

There are following assignment operators supported by C++ language −

Misc Operators

The following table lists some other operators that C++ supports.

2. Differentiate between Object Oriented Approach and functional programming.

1. Functional programming is used for performing many different operations for which the data is fixed. Object-oriented programming used for performing few operations which are having common behavior and different variants.
2. Functional programming is having a stateless programming model. Object-oriented programming is having a stateful programming model.
3. In functional programming, a state does not exist. In object-oriented programming, the state exists.
4. In functional programming, a function is the primary manipulation unit. In object-oriented, an object is the primary manipulation unit.
5. In functional programming, its functions have no side effects means does not make any impact on code that is running on multiple processors. In Object-oriented programming, its methods can have side effects and may put an impact on processors.
6. In functional programming, the main focus of programming is *what are we doing*. In object-oriented programming, the main focus of programming is *how are we doing*.
7. Functional programming mainly supports abstraction over data and abstraction over behavior. Object-oriented programming mainly supports abstraction over data only.
8. Functional programming provides high performance in processing the large data for the applications. Object-oriented programming is not good for big data processing.

[III] Long Answer Questions (Max 150 words). **(2*10=20)**

1. Discuss Object Oriented along with its features and advantages.

Object oriented programming language (OOPL) is a high level programming language based on objected oriented programming model (OOPM). Object oriented programming comprises of logical classes, objects, methods and relationships.
Object oriented programming is a programming language that focuses on objects (i.e. data) rather than logic (i.e. procedures). If we talk about structured programming language then it focuses on procedures and not on the data. So in object oriented programming we focus on how we can manipulate the object rather than logic which are used to manipulate data of the object. Each object is independent of each other and executed independently. It can execute by itself and can be interchanged with other objects. Object interacts by interchanging information with each other.

Advantages of Object Oriented Programming

1. Objects are created on real world entities.
2. Through object oriented programming language, complex systems of real world can be converted into software solutions.
3. Large programs are difficult to develop. Objected oriented programs focuses the developer to do extensive planning which will reduces programming flaws and better design.
4. Software maintenance of the object oriented programming is easier than as compared with structured oriented programming as much time has been invested while doing the planning of the code so minimum flaws are found.
5. Object oriented programming can be easily reused in other programs due to its re-usability feature.
6. Objected oriented programming is easy to implement.
7. Object oriented programming can easily extensible. New features or changes in operating environment can be easily done.

Basic features of object oriented programming language:

Encapsulation

Encapsulation is a concept in object oriented programming which focuses on protecting the variable and function from outside the world in order to manage the piece of code in better way and produce least impact on other part of the code.

Abstraction

Abstraction is the process of implementing the essential features and removing unnecessary information. Abstraction is an important characteristic in which the programmer hides the irrelevant details which will in turn reduces the size of the code and the complexity of the code. Abstraction makes things more general, simpler and abstract.

Inheritance

Inheritance is another main feature of object oriented programming language. Inheritance allows the class to use the property and method of other class. In other words, we can say that the sub classes derive the property and functionality from the base class. Subclass is also known as derived class and base class is known as super class.

Polymorphism

Polymorphism is way by which we are defining multiple functionalities under the same names. In other words we can say that we have the same code or operation but it will behave differently in different contexts.

2. Write a C++ program to find the sum and average of 10 numbers entered by user.

```cpp
#include <iostream>
using namespace std;

int main()
{
   int n, i;
   float num[100], sum=0.0, average;

   cout << "Enter the numbers of data: ";
   cin >> n;

   while (n > 100 || n <= 0)
   {
      cout << "Error! number should in range of (1 to 100)." << endl;
      cout << "Enter the number again: ";
      cin >> n;
   }

   for(i = 0; i < n; ++i)
   {
      cout << i + 1 << ". Enter number: ";
      cin >> num[i];
      sum += num[i];
   }

   average = sum / n;
   cout << "Average = " << average;

   return 0;
}
```