CHAPTER-2

OBJECT ORIENTED PROGRAMMING

VERY SHORT/ SHORT ANSWER QUESTIONS

1. Discuss major OOP concepts briefly.

Ans. Following are the general OOP concepts:
1. Data Abstraction: Data abstraction means, providing only essential information to the outside word and hiding their background details i.e. to represent the needed information in program without presenting the details.
2. Data Encapsulation: The wrapping up of data and operations/functions (that operate o the data) into a single unit (called class) is known as Encapsulation.
3. Modularity: Modularity is the property of a system that has been decomposed into a set of cohesive and loosely coupled modules.
4. Inheritance: Inheritance is the capability of one class of things to inherit capabilities or properties from another class.
5. Polymorphism: Polymorphism is the ability for a message or data to be processed in more than one form.

2. What are programming paradigms? Give names of some popular programming paradigms.

Ans. Programming Paradigm: A Programming Paradigm defines the methodology of designing and implementing programs using the key features and building blocks of a programming language.
Following are the different programming paradigms:
(i) Procedural Programming
(ii) Object Based Programming
(iii) Object Oriented Programming

3. What are the shortcomings of procedural and modular programming approaches?

Ans. Following are the various shortcomings of procedural and modular programming approaches:
- ✓ Procedural Programming is susceptible to design changes.
- ✓ Procedural Programming leads to increased time and cost overheads during design changes.
- ✓ Procedural and Modular programming both are unable to represent real world relationship that exists among objects.
- ✓ In modular programming, the arrangement of the data can't be changed without modifying all the functions that access it.

4. Write a short note on OO programming.

Ans. OOP stands for Object Oriented Programming. In, Object-Oriented Programming (OOP), the program is organized around the data being operated upon rather than the operations performed. The basic idea behind OOP is to combine both, data and its functions that operate on the data into a single unit called object.
Following are the basic OOP concepts:
1. Data Abstraction   2. Data Encapsulation   3. Modularity
4. Inheritance   5. Polymorphism

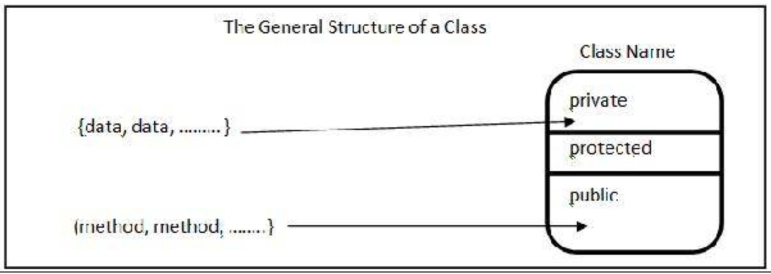
5. How does OOP overcome the shortcomings of traditional programming approaches?

Ans. OOP provides the following advantages to overcome the shortcomings of traditional programming approaches:
- ✓ OOPs is closer to real world model.
- ✓ Hierarchical relationship among objects can be well-represented through inheritance.
- ✓ Data can be made hidden or public as per the need. Only the necessary data is exposed enhancing the data security.
- ✓ Increased modularity adds ease to program development.
- ✓ Private data is accessible only through designed interface in a way suited to the program.

6. Write a short note on inheritance.

Ans. Inheritance enables us to create new classes that reuse, extend, and modify the behavior that is defined in other classes. The class whose members are inherited is called the base class, and the class that inherits those members is called the derived class. A derived class can have only one direct base class. Inheritance is transitive.
When we define a class to derive from another class, the derived class implicitly gains all the members of the base class, except for its constructors and destructors.

7. What are the advantages offered by inheritance?

Ans.	✓ Inheritance ensures the closeness with the real-world models. ✓ Allows the code to be reused as many times as needed. The base class once defined and once it is compiled, it need not be reworked. ✓ We can extend the already made classes by adding some new features. ✓ Inheritance is capable of simulating the transitive nature of real-world's inheritace, which in turn saves on modification time and efforts, if required.
8.	**Do you think OOP is more closer to real world problems? Why? How?**
Ans.	Yes, OOP is more closer to real world problems because object oriented programming implement inheritance by allowing one class to inherit from another. Thus a model developed by languages is much closer to the real world. By implementing inheritance, real-world relations among objects can be represented programmatically.
9.	**How are classes and objects implemented in C++?**
Ans.	A class is a blueprint for object. A class is implementing with the help of an object. Suppose a class has a member function named display(). Now, to implement that member function display we have to use object of that class. The objects is implemented in software terms as follows: (i) characteristics / attributes are implemented through member variables or data items of the object. (ii) behavior is implemented through member functions called methods. (iii) It is given a unique name to give it identify.
10.	**What is the significance of private, protected and public specifiers in a class?**
Ans.	A class groups its members into three sections: private, protected, and public. The private and protected members remain hidden from outside world. Thus through private and protected members, a class enforces data-hiding. The public members are accessible everywhere in a program.  The General Structure of a Class Class Name private protected public {data, data, ...} {method, method, ...}
11.	**While implementing encapsulation, abstraction is also implemented. Comment.**
Ans.	Abstraction is the act of representing essential features without including the background details. Encapsulation is the way of combining both data and the functions that operate on the data under a single unit. Encapsulation is the way of implementing abstraction. Thus, it is true that while implementing encapsulation, abstraction is also implemented.
12.	**Discuss the relation between abstract and concrete classes.**
Ans.	In C++ an *Abstract Class* is the one, which defines an interface, but does not necessarily provide implementation for all its member functions. An abstract class is meant to be used as the base class from which other classes are derived. The derived class is expected to provide implementations for the member functions that are not implemented in the base class. A derived class that implements all the missing functionality is called a *Concrete Class.* A concrete class derives from its abstract class.
13.	**Illustrate the concept of Inheritance with the help of an example.**

Ans.	Inheritance is the capability of one class to inherit properties from another class. For instance, *Student* is a class wherefrom class *GraduateStudent* inherits as given below:
	`class Student`
	`{`
	`      protected:`
	`            char name[25];`
	`            int rollno;`
	`      public:`
	`           void readStudent();`
	`           void showStudent();`
	`};`

```
class GraduateStudent:protected Student
{
     protected:
           char subjects[50];
     public:
          void readAradStud();
          void showGradStud();
};
```

14. **Encapsulation is one of the major properties of OOP. How is it implemented in C++?**

Ans. A class binds together data and its associated functions under one unit thereby enforcing encapsulation as encapsulation means wrapping up data and associated functions together into a single unit. While implementing encapsulation, following things are taken care of:

(i) Anything that an object does not know or cannot do is excluded from the objects.
(ii) Encapsulation is used to hide unimportant implementation details from other objects.
(iii) The data and associated functions are wrapped up in one unit called class.

Through encapsulation, an interface is made available to world through which users can use the class.

15. **Reusability of classes is one of the major properties of OOP. How is it implemented in C++?**

Ans. Reusability of classes is implemented through inheritance in C++. Inheritance is implemented by specifying the name of the (base) class from which the class being defined (the derived class) has to inherit from.
It is done with the following syntax:

```
        class<derived class name> : <base class name>
        {
                <- derived class own features.
        }
```

16. **How is polymorphism implemented in C++?**

Ans. C++ implements polymorphism through virtual functions, overloaded functions and overloaded operators.
The term 'overloaded function' refers to a function having one name and more than one distinct meaning.
For example,

```
     float area(float a)
     {
          return a*a;
     }
     float area(float a,float b)
     {
          return a*b;
     }
```

TYPE C : LONG ANSWER QUESTIONS

1. **Write briefly about different programming paradigms.**

Ans. Programming Paradigm: A Programming Paradigm defines the methodology of designing and implementing programs using the key features and building blocks of a programming language. Following are the different programming paradigms:
(i) Procedural Programming: Procedural programming paradigm lays more emphasis on procedure or the algorithm. Data is considered secondary. Data is loosely available to many functions. This paradigm is: Decide which procedure we want; use the best algorithm we can find.
(ii) Object Based Programming: In object based programming, data and its associated meaningful functions are enclosed in one single entity a *class*. Classes enforce information hiding and abstraction thereby separating the implementation details and the user interface. It also supports user-defined types.
(iii) Object Oriented Programming: Object oriented programming offers all the features of object based programming and overcomes its limitation by implementing inheritance. The object oriented approach views a problem in terms of objects involved rather than procedure for doing it. We can define the object oriented programming (OOP) paradigm as following: Decide which classes and objects are needed; provide a full set of operations for each class.

2. **Discuss OOP concepts. How are these implemented in software terms in C++?**

Ans.	Following are the general OOP concepts: **1. Data Abstraction:** Abstraction refers to the act of representing essential features without including the background details or explanations. Abstraction is implemented through public members of a class, as the outside world is given only the essential and necessary information through public members, rest of the things remain hidden. **2. Data Encapsulation:** The wrapping up of data and operations/functions (that operate o the data) into a single unit (called class) is known as Encapsulation. Encapsulation is implemented with the help of a class as a class binds together data and its associated function under one unit. **3. Modularity:** The act of partitioning a program into individual components is called modularity. C++ implements modularity through separately compiled files. The traditional practice in the C++ community is to place module interface in files named with a ***.h*** suffix; these are called header files which can be used through #include directive. **4. Inheritance:** Inheritance is the capability of one class of things to inherit capabilities or properties from another class. Inheritance is implemented in C++ by specifying the name of the (base) class from which the class being defined (the derived class) has to inherit from. **5. Polymorphism:** Polymorphism is the ability for a message or data to be processed in more than one form. C++ implements polymorphism through virtual functions, overloaded functions and overloaded operators.