Inheritance

One Mark Questions

Question 1.
What is inheritance?
Answer:
In object-oriented programming, inheritance is a way to form new classes using classes that have already been defined.

Question 2.
How to implement inheritance?
Answer:
The inheritance is implemented using the following syntax class derived_classname : access specifier base classname.

Question 3.
What is base class?
Answer:
A base class is a class from which other classes are derived.

Question 4.
What is derived class?
Answer:

- The class that inherits is called derived class.
- The inheriting class is called the derived class.

Question 5.
What is public access specifier?
Answer:
It means "public parts of base class remain public and protected parts of base class remain protected for derived class".

Question 6.
What is private access specifier?
Answer:
The private access specifier means "Public and Protected Parts of base class remain Private in derived class".

Question 7.
Mention any one advantage of inheritance? .

Answer:
The one advantage of inheritance is reusability.

Question 8.
Is inheritance possible in c?
Answer:
No, because it doesn't support object oriented concept.

Question 9.
What is the use of protected access specifier?
Answer:
The private member of a base class cannot be inherited to the derived class and protected member of a base class remains protected in a derived class.

Question 10.
What is the use of public access specifier?
Answer:
The private members of a base class cannot be inherited and protected members remain protected in a derived class.

Question 11.
What is the use of private access specifier?
Answer:
The private members of a base class cannot be inherited and protected members of a base class become private members to derived class.

Question 12.
What is single inheritance?
Answer:
It is a derived class with only one base class is called single inheritance

Question 13.
What is multilevel inheritance?
Answer:
A class can be derived from another derived class which is known as multilevel inheritance.

Question 14.
What is hierarchical inheritance?
Answer:
When the properties of one class are inherited by more than, one class, it is called hierarchical inheritance.

Question 15.
What is hybrid inheritance?
Answer:
It is the combination of hierarchical and multilevel inheritance.

Question 16.
What is mulitple inheritance?
Answer:
When a class inherit properties from more than one class is known as multiple inheritance

Question 17.
What is virtual base class?
Answer:
The base class is declared as the virtual base class so that only one copy of its members are inherited by the derived class in a hybrid inheritance.

Question 18.
What is an abstract class?
Answer:
Abstract class is one that is not used to create objects. An abstract class is designed only to act as a base class.

Question 19.
When do we need to inherit?
Answer:
Suppose X is a class exist and we need new class Y having all properties of X and in addition to its own, then we need to inherit X class to derive Y class.

Question 20.
When is it necessary to use inheritance?
Answer:
When a newly created class need to acquire properties of existing class and with its own properties then it is necessary to use inheritance.

Question 21.
What is visibility mode?
Answer:
The visibility mode in inheritance specify the access mode of base class members to derived class.

Two Mark Questions

Question 1.
How to implement inheritance?
Answer:
The inheritance is implemented using the following syntax
class derived_class name : access specifier base class name

Question 2.
What is the difference between public and private access specifier?
Answer:
In public access mode, protected members of base class remain protected in a derived class. In private access mode, protected members of a base class become private members in a derived class. And the private members of a base class cannot be inherited in both the access mode.

Question 3.
Mention any two advantages of inheritance?
Answer:

- Reusability- Inheritance helps the code to be reused in many situations.
- Saves Time and Effort- Since the main code written can be reused in various situations as needed.

Question 4.
Mention any two types of inheritance.
Answer:

- Single inheritance: A derived class with only one base class is called single inheritance
- Multiple inheritance: A class can inherit properties from more than one class which is known as multiple inheritance.

Question 5.
What is the difference between inheritance and polymorphism?
Answer:
The concept of inheritance provides the idea of reusability. This means that we can add additional features to an existing class without modifying it. Polymorphism means that if the same message is sent to different objects, the object's behavior depends on the nature of the object it self This means that a general class of operations may be accessed in the same manner even though specific action associated with each operation may differ.

Question 6.
What is single inheritance? Give an example.

Answer:
Single inheritance:
A derived class with only one base class is called single inheritance. For example, If A is base class then class B derive from base class A.

Question 7.
What is multilevel inheritance? Give an example.
Answer:
A class can be derived from another derived class which is known as multilevel inheritance. For example, The derived class C inherit B class whereas B is derived from class A.

Question 8.
What is hierarchical inheritance? Give an example.
Answer:
When the properties of one class are inherited by more than one class, it is called hierarchical inheritance. For example, class B, C and D are derived from base class A.

Question 9.
What is hybrid inheritance? Give an example.
Answer:
It is the combination of hierarchical and multilevel inheritance. For example, The derived class D derive from the classes B and C whereas both the classes B and C are derived from base class A.

Question 10.
What is multiple inheritance? Give an example.
Answer:
A class can inherit properties from more than one class which is known as multiple inheritance. For example, the class C derive from base class A and base class B.

Question 11.
What is virtual base class? Give example.
Answer:
The base class is declared as the virtual base class so that only one copy of its members are inherited by the derived class in a hybrid inheritance. For example, the derived class D derive from the classes B and C whereas both the classes B and C are derived from base class A. Then base class A is inherited twice and to avoid two copies of class A to class D, class A is declared virtual base class.

Question 12.
What is an abstract class?
Answer:

The class is not used to create objects are called abstract class. It is used as base class to get derive class only.

Question 13.
Which are the components which cannot be inherited?
Answer:
A derived class inherits every member of a base class except:

- its constructor and its destructor
- its operator=() members
- its friends

Question 14.
Explain single inheritance with a suitable C++ program.
Answer:

```
class studentinformation
{
        private:
                int rollno;
                char  name[25];
        public:
                void readinformation()
                {
                    Cout << "Enter the roll number and name of the student " << endl;
                    Cin >> rollno >> name;
                }
                void printinformation()
                {
                    cout << "The roll number : " << rollno << endl;
                    cout << "The name of the student : " << name << endl;
                }

};
```

```
class studentmarks : public studentinformation
{
        private:
        int m1, m2, total;
        public:
        void readmarks()
        {
                cout<< "Enter the two marks " << endl;
                cin >> m1 >> m2;
        }
        void printmarks()
        {
                total = m1 + m2;
                cout << "Marks 1 : " << m1 << endl;
```

Question 15.
Explain multilevel inheritance with a suitable C++ program
Answer:

```
class studentparents
 {
        private:
                char  fname[25];
                char mname[25];
        public:
                void readparents()
                {
                   cout << "Enter the Father's name and Mother's name " << endl;
                   cin >> fname >> mname;
```

```
        }
        void printparents()
        {
            cout << "Fathers name : " << fname << endl;
            cout << "Mother's name : " << mname << endl;
        }

};

class studentinformation : public studentparents
{
    private:
        int rollno;
        char  name[25];
    public:
        void readinformation()
        {
            Cout << "Enter the roll number and name of the student " << endl;
            Cin >> rollno >> name;
        }
        void printinformation()
        {
            cout << "The roll number : " << rollno << endl;
            cout << "The name of the student : " << name << endl;
        }

};
```

```
class studentmarks : public studentinformation
{
        private:
        int m1, m2, total;
        public:
        void readmarks()
        {
                cout<< "Enter the two marks " << endl;
                cin >> m1 >> m2;
        }
        void printmarks()
        {
                total = m1 + m2;
                cout << "Marks 1 : " << m1 << endl;

              cout << "Marks 2 : " << m2 << endl;
              cout << "Total Marks  : " << total << endl;
         }
};

void main()
{
        studentmarks  std;
        std.readparents ();
        std.readinformation();
        std.readmarks();

        std.printparents();
        std.printinformation();
        std.printmarks();
}
```

Question 16.
Explain multiple inheritance with a suitable C++ program.
Answer:

```
class date
{
        protected:
        int dd, mm, yy;
        public:
        void readdate()
        {
        cout << "Enter day, month and year" << endl;
        cin >> dd >> mm >> yy;
        }
        void printdate()
        {
        cout << dd << ":" << mm << ":" << yy << endl;
        }
};

 class time
        protected:
        int  hr, mn, sec;
        public:
        void readtime()
        {
        cout << "Enter hours, Minutes and second" << endl;
        cin >> hr >> mn >> sec;
```

```
  }
  void printtime()
  {
  cout << hr << ":" << mn << ":" << sec << endl;
  }
  };
class datetime : public date, public time
{
        public:
        void readdatetime()
        {
                date ::readdate();
                time::readtime();
        }
        void printdatetime()
        {
                date :: printdate();
                time :: printtime();
        }
};
```

Question 17.
How does inheritance influence the working of constructor?
Answer:
The inheritance influence the working of constructor, if base class contains parameterized constructor, then it is compulsory to pass the arguments using derived class. Normally derived class is used to declare objects therefore derived class is used to pass arguments to base class constructor.

If both base class and derive class has constructors, then base class constructor is executed first and then derived class constructor is executed i.e., pass the value using derived class to base class constructor and then to derived class constructor. The derived class is responsible to pass arguments to both base class as well as derived class.

Question 18.
How does inheritance influence the working of destructors?
Answer:

- Constructors, destructors and overloaded assignment operators are not inherited.
- C++ requires that a derived-class constructor call its base-class constructor to initialize the base-class data members that are inherited into the derived class.
- If the derived-class's constructor did not invoke the base-class's constructor explicitly, C++ would attempt to invoke the base class's default constructor.
- When an object of a derived class is built(constructed):
- the base-class's constructor is called immediately to initialize the base-class data members in the derived-class object and then
- the derived-class constructor will initialize the additional derived-class data members.
- When a derived-class object is destroyed, the destructors are called in the reverse order of the constructors:
- First the derived-class destructor is called, then the base-class destructor.
- When deriving a class, the base class may be declared as either public, protected, or private.