Assignment - 16

Q) Explain the concept of inheritance & what are the types of inheritance according to the architecture

→ **Inheritance**

- Inheritance is considered as one of the object oriented paradigm of c++
- In simple terms inheritance is defines reusability.
- By using the concept of inheritance class will Acquire characteristics & behaviours of another class
- In Inheritance the parent & child or Base and derived class are present

syntax for inherit the class

Derived class : Access specifier Base class

- following are the types of inheritance

1) Single level inheritance
2) multilevel inheritance
3) multipal inheritance
4) ~~Hier~~

Q2] Explain the Concept of Access specifier in detail

→ The Access Specifier in C++

- The Access specifier indicates who can Access & who cannot Access the member of class

- in C++ programming there are 3 Access specifier

1) Public Access specifier
2) private Access specifier
3) protected Access specifier

- The public access specifier is used to specify that a class member can be accessed from Anywhere. both inside & outside the class.

- The private Access specifier is used to specify that a class member can only be accessed from within the class. this means that any function outside the class cannot access the private members of the class.

- Private Access specifier is used in Abstraction.

- The protected access specifier is used to specify that a class member can be accessed from within the class & its derived class this means that any function or object outside the class hierarchy cannot access the protected members of the class.

Q3] What is the difference between private and protected Access Specifier

Private Access specifier	Protected Access specifie
1] The class members declared as private can be Accessed only by the functions inside the class	1] Protected class mem ber declared as Protected are inaccessible outside the class but they can be accessed by any subclass of the class
2] Private Access specifier is used in Abstraction	2] Protected Access Specifier is used in inheritance
3] Private members can also Accessed through the friend function	3] Protected members cannot be accessed through the friend function
4] Nimited to the same class	4] Accessable within the Same class & derived class

Q4) What is the default access specifier if it is not written?

→ If we do not specify any access specifiers for the members inside the class then by default the access specifier for the members will be private.

- The private members are not accessible outside the class they can be accessed only through member functions of the class.

Q5) What are the types of inheritance according to access specifiers?

→ 1) Public inheritance makes public members of the base class public in the derived class and the protected members of the base class remain protected in the derived class

- protected inheritance makes the public and protected members of the base class protected in the derived class

- Private inheritance makes the public and protected members of the base class private in the derived class.

06] Explain the constructor and destruct calling sequence in case single level, multilevel and multiple inheritance

→ Single level inheritance:

- In the single level inheritance the constructor of parent (base) class is called first, followed by child (derived) class constructor.
- The destructor will be called in the opposite order.

Multilevel inheritance

- The constructor of the topmost (grandparent superclass) called first, followed by constructor of the parent & child classes.
- The destructor is called in the reverse order of the constructors.

Multiple Inheritance

- The constructors of the base classes are called in the order they are specified in the program.
- The destructor is called in the reverse order of the constructors.

Q7] Draw object layout and class diagram of below code snippets and explain its internal working in detail. Explain the type of inheritance in the below code snippet

```
class base
{
  public:
    int i;
    float f;
    double d;

    void fun()
      { }

    void gun()
      { }
};

class Derived : public base
{
  public:
    int i;
    double d;

    void sun()
      { }
};

int main()
{ base bobj;
  Derived dobj;
  return 0; }
```

PAGE No.
DATE / /

object layout

bobj

100		base::i
104		base::f
108		base::d
116		

class Diagram of Base class

characteristics
i, f, d
Behaviours
void fun()
gun()

dobj

100		Derived::i
104		Derived::d
112		base::i
116		base::f
120		base::d
128		

class Diagram of Derived class

characteristics
i, d
Behaviours
gun()

- In the above program the Base which having characteristics int i, float f, double and the Behaviours of fun() & gun()

- class Derived inherit the base class which acquire the all properties of Base class.

- bobj is the object of Base class using that object we can Access the Behaviours & characteristics of Base class.

- dobj is the Der obj of Derived class which using that object we can Access the properities of Base class as well as child ~~ch~~ class.

- Single level inheritance is used in the above code.

- In the Single level inheritance in which a single derived class is inherited from a single base class

- In the Single level inheritance the parent (base) child (derived) class is present.

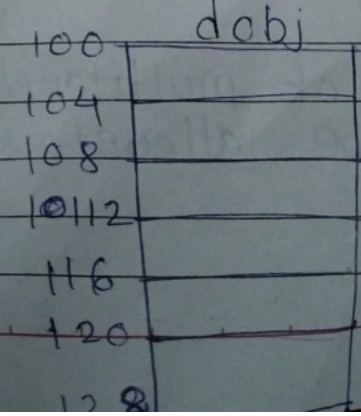
Q8) Draw object layout and class diagram of below code snippets and explain its internal working in detail. Explain the type of inheritance in the below code snippet.

→

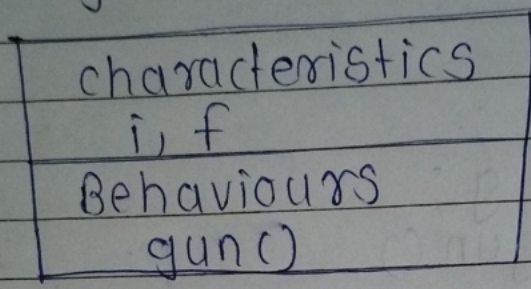
```
class base1
{
    public:
        int i;
        float f;
        void gun()
        {}
};
```

```
class base2
{
  public:
      int j;
      float g;
      void gun()
        {}
};

class Derived : public base1, base2
{
  public:
      int i;
      double d;
      void sun()
        {}
};

int main()
{
  Derived dobj;

  return 0;
}
```

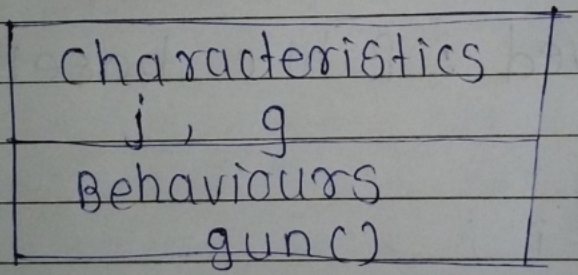
object layout



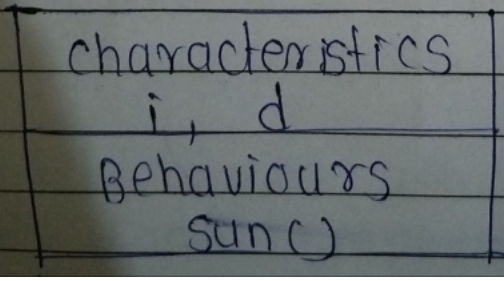
class diagram of base1 class



class diagram of base2 class



class diagram of Derived class



PAGE No.
DATE / /

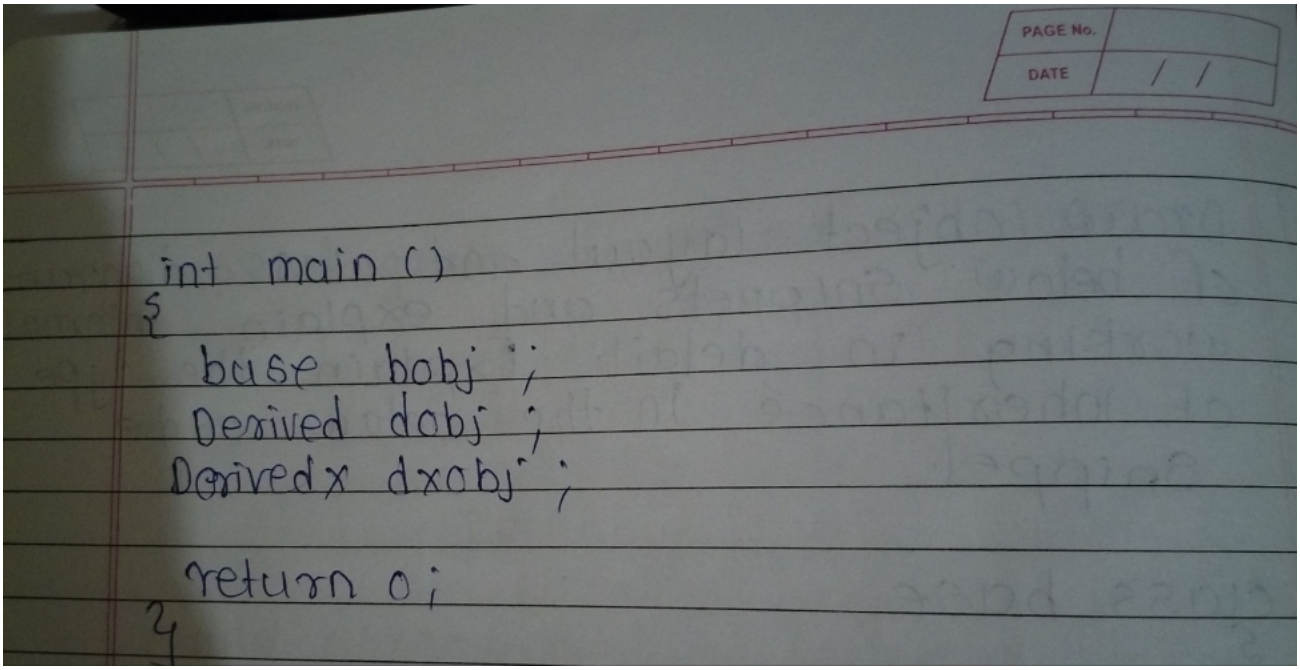
Multipale inheritance

- In this type of inheritance one class can inherit more than one classes at a time.

- multiple ~~classe~~ inheritance in C++ allows a derived class to inherit more ~~properties~~ & characteristics of Behaviours of multiple base classes

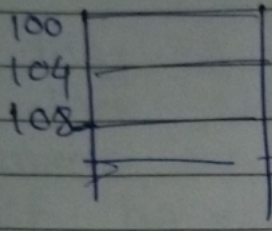
Q7] Draw object layout and class diagram of below snippets and explain internal working in detail. Explain the type of inheritance in the below code snippet

```
class base
{
  public:
      int i;
     float f;
     void fun()
        {}
     void gun()
        {}
};

class Derived : public base
{
  public:
      int i;
     double d;

     void sun()
        {}
};

class Derivedx : public Derived
{
   public:
      int k;
     void run()
        {}
```

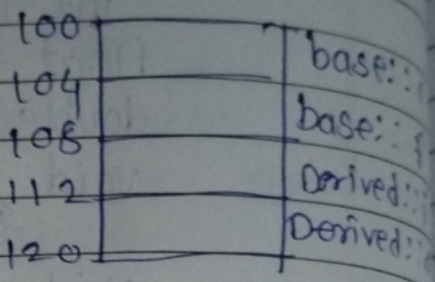

PAGE No.
DATE
/ /
int main ()
{
base bobj ;
Derived dobj ;
Derivedx dxobj ;
return 0;
}

object layout

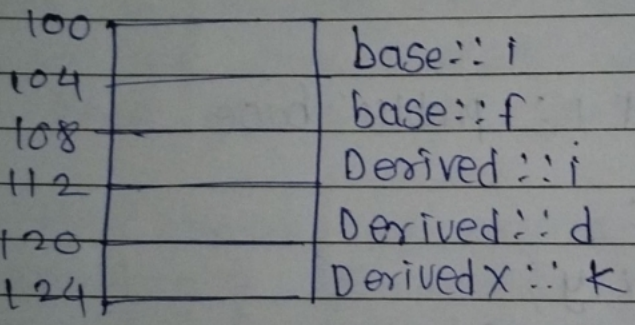
bobj



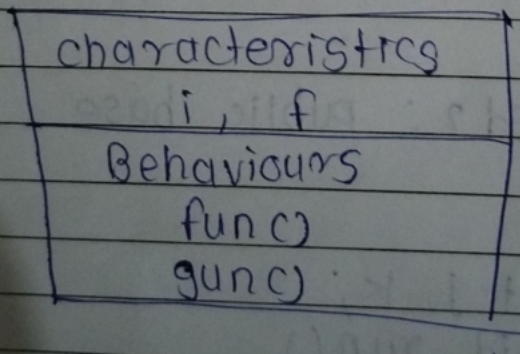
dobj



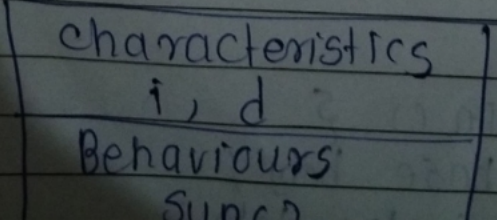
dxobj



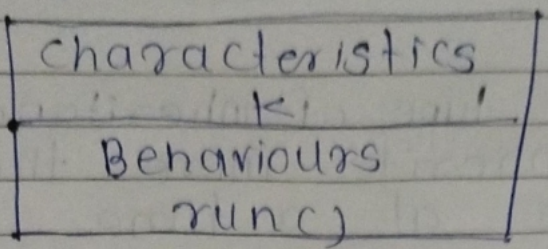
Class Diagram of base class



Class Diagram of Derived class



class Diagram of Derivedx class



- In the above program base class, Derived, Derivedx classes are present

- Derived class is inherited by base class means Derived class obje has access of Derived & base cla

- Derived 2 class is inherited the Derived class means Derived 2 class has Access of base & Derived class.

Multilevel Inheritance

- Multilevel inheritance used in this program.

- In case of multilevel inheritance there should be atleast 3 classes.

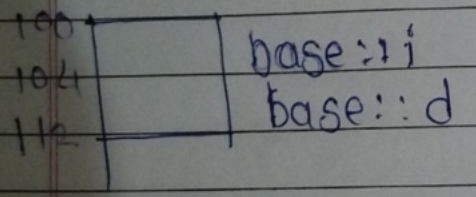
- In multilevel inheritance Grandfather father and child relationship.

Q10)

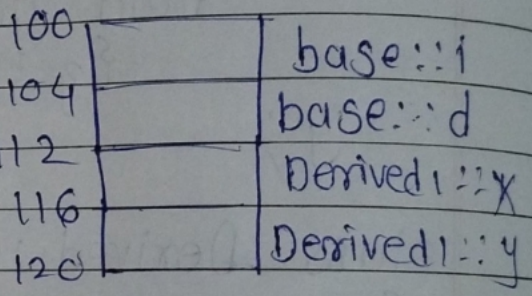
```
class base
{
  public:
     int i;
     float f;
     double d;
     void fun()
        {}
     void gun()
        {}
};

class Derived1 : public base
{
   public:
     int x,y;
     void sun()
        {}
};

class Derived2 : public base
{
   public:
     int j, k;
     void run()
        {}
};
int main() {
     base baby;
     Derived1 d obj;
     Derived2 dobj2;
     return 0;
```

object layout

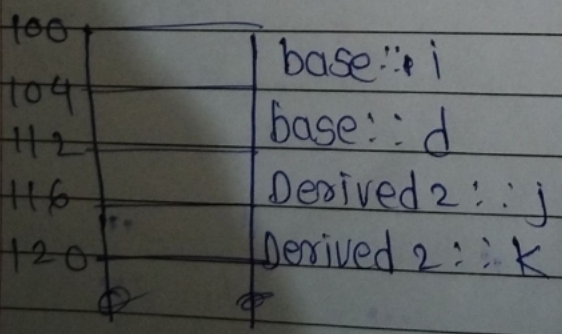
bobj



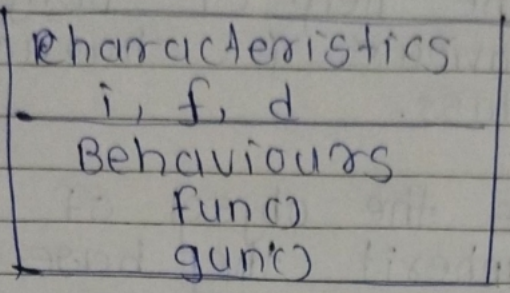
dobj



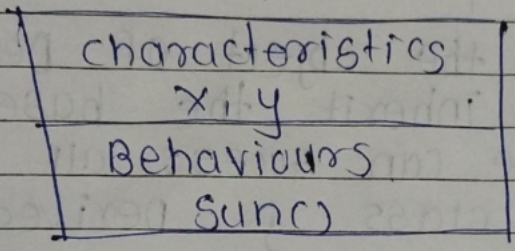
dobj2



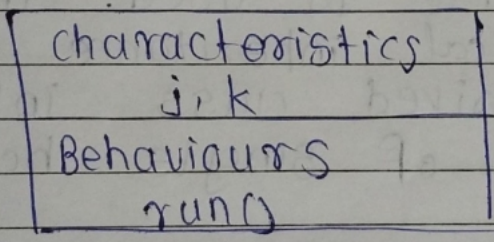
class diagram of base class



class diagram of Derived1 class



class diagram of Derived2 class



- in the above code there are 3 class i.e base class, Derived1 class & Derived 2 class
- In the program Derived1 class inherit the base class
- Derived 2 class also inherit the base class.

- bobj is the object of base class with the help of bobj we Access All behaviours & characteristics of base class.

- dobj is the object of Derived1 class which inherit the base class that mean we can Access all properties of base class & Derived1 class

- dobj2 is the object of Derived2 class which inherit the base class that mean we can Access all properties of base class & Derived2 class.

- In the above program Hierarchical inheritance is used. in the hierarchical inheritance more than one derived class inherit the property of single base class.