Page No.:
Date: 09/10/25

Assignment -16

1) What are the types of inheritances according to the architecture?

There are ~~four~~ three types of inheritances according to architecture:

2) Single - Level Inheritance

In single-level inheritence one classs inherits from its base ~~cla~~ class its characteristic & behaviours

Example:

```
Class Demo {                    // Parent classs
    public:
        int x, y;
        Demo() {
            cout<<" Inside Demo constructor "<<endl;
            x=10;
            y=20;
        }
        ~Demo() {
            cout<<" Inside Demo Destructor "<<endl;
        }
        void Fun() {
            cout<<" Inside fun of Demo"<<endl;
        }
};

Class Hello : public Demo       // Derived class (child)
{
```

```
public:
    int a, b;
    Hello() {
    cout << "Inside Hello constructor" << endl;
      a = 50;
      b = 60;
    }
   ~Hello()
    {
      cout << "Inside Hello destructor" << endl;
    }
   void fun() {
      cout << "Inside Hello fun" << endl;
    }
};
int main()
{
    Hello hobj;
    cout << "size of hello object is" << sizeof(hobj) << endl; // 16 bytes

  cout << hobj.x << endl;   // 10
  cout << hobj.y << endl;   // 20
  cout << hobj.a << endl;   // 50
  cout << hobj.b << endl;   // 60

  hobj.fun();
  hobj.gun();
   return 0;
}
```

OUTPUT

```
16 bytes
Inside Demo constructor
Inside Hello constructor
16 bytes
10
20
50
60
Inside Hello destructor
Inside Demo destructor
```

Multi-Level Inheritance:

1. It is considered as extension of single level inheritance
2. In case of Multi-level inheritance, there should be at-least 3 classes
3. Example

```
class Base {
    public:
    int a,b;
    Base()
      {
      }
    void Display(){
      }
};

class Derived1:: public Base
{
    public:
      int x,y;
      Derived1()
      {
      }
      Void display1(){
      }
};

Class Derived2:: public Derived1
{
    public:
      int p,q;
```

```
    Derived2(){
    }
    void display2()
    {
    }
};

int main()
{
    Derived2 dobj;
    cout<<"size of object is: "<<sizeof(dobj)<<endl; //24
    return 0;
};  cout<<dobj.a<<endl;
    cout<<dobj.b<<endl;
```

3] Multiple:-Level inheritance:

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

```
class Demo1
{
    int a,b;
    Demo1(){
    }
    void display1(){
    }
};

Class Demo2
{
    int x,y;
```

```
Demo2() Demo2 (){
        }
        void Display2(){
        }
};

class Derived : public Demo1, public Demo2
{
    public:
        int p,q;
        Derived(){
        }
        void Display3 (){
        }
};

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

Explain the concept of Access specifiers in detail?

The term access specifiers indicates, who can access & who cannot access the members of a class.

In C++ Programming, there are three access specifiers

1] Public
2] Private
3] Protected

Access Specifier	Inside same class	Outside class	Inside main	Inside Derived class	Inside other class
Private	Allowed	NA	NA	NA	NA
Protected	Allowed	NA	NA	Allowed	NA
Public	Allowed	Allowed	Allowed	Allowed	Allowed

* Irrespective of access specifiers memory for all characteristics gets allocated.

Q 3] What is difference between private & protected access specifier?

Ans The only difference between private & protected is:

1] Private : Private members cannot be accessed inside,

~~1] Outside class~~

~~2] Inside main~~

3] Inside Derived class

~~4] Inside other class~~

3] Protected : Protected members can be accessed derived class.

Q 4] What is the default access specifier if it is not written explicitly?

Ans If ~~default~~ access specifier is not ~~written~~ explicitly, default access specifier is private.

5] Explain the constructor & Destructor calling sequence in case of single level, multi level & multiple inheritance.

1] Single - Level Inheritance:

- When we create an object of derived class first the constructor of base class gets called & then constructor of derived class gets called.
- Constructor calling sequence is top to bottom & the destructor calling sequence is from bottom to top

Base constructor
Derived Constructor

Derived Destructor
Base Destructor

2] Multiple - Level Inheritence

- Constructor & Destructor sequence is same as Single - Level Inheritance.

Base constructor
Derived1 constructor
Derived2 constructor

Derived2 Destructor
Derived1 constructor
Base Destructor

PAGE No.
DATE / /

3] Multiple - Level Inheritance

Base 1 Constructor
Base 2 Constructor
Derived constructor
:
Derived Destructor
Base 2 Destructor
Base 1 Destructor

Q5] What are the type of inheritances according to access specifiers?

Ans According to :-

1] Private : Private ~~properties~~ characteristics can not be inherited there no inheritance takes place in case of private access specifier.

2] Protected : Protected ~~pr~~ characteristics can be inherited to the derived class only there is single level inheritance, multi-level inheritance & multiple inheritance

3] Public : Public characteristics can be ~~ext~~ accessed by anyone from outside class, inside class, in the main, inside other class & from anywhere in the code.

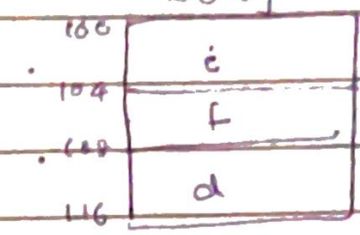
Page No.
Date: / / /

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

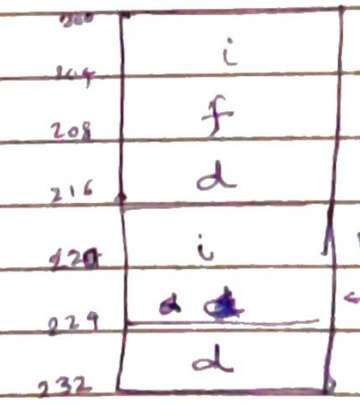
```
class Base
{
  pubic
    int i;
    float f;
    double d;
    void fun()
    {}
    void gun()
    {}
};
class Derived : public base
{
  public;
    int i;
    double d;
    void sun()
    {}
}
int main()
{
  base bobj;
  derived dobj;
  return 0;
}
```

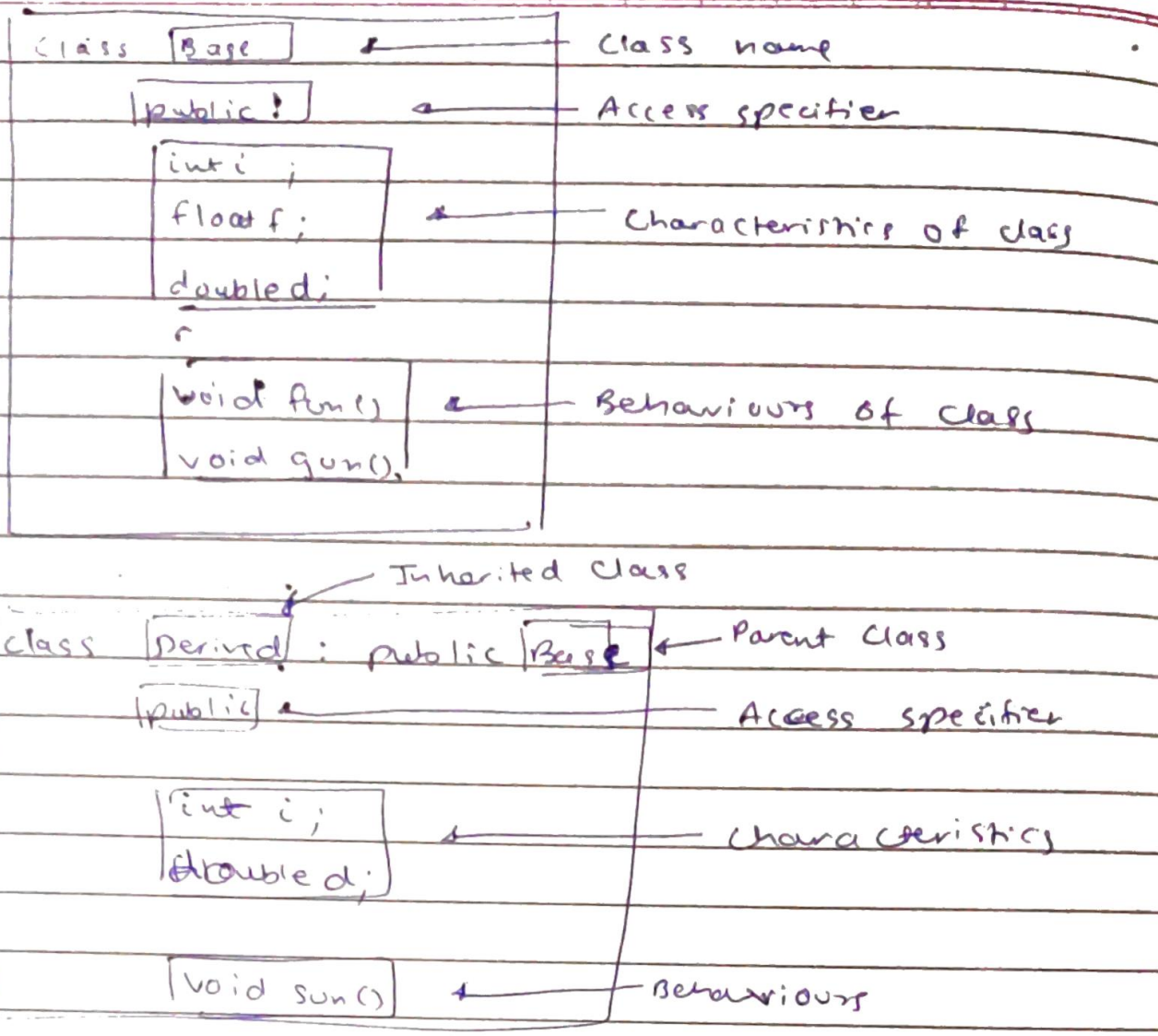
object layout

bobj



dobj





1] The above code use single Level Inheritance

2] In single level Inheritance one class inherits characteristics & behaviours of its base class

3] In the above example Base class is the parent class whose characteristics and behaviours are public & accessible from anywhere in the code.

4] Derived class inherits the ~~to~~ characteristics & behaviours of class Base. This means an objects of derived class can access characteristics & behaviours of base class also

5] ~~Inher~~ Derived class has its own ~~properties also~~ characteristics & behaviours which cannot be accessed by Base class object.

∴ class Derived padding gets added due to memory allocation in terms of highest member i.e. double d of derived class.

∴ Total object size will be 32 bytes instead of 28

∴ i.e. return 0 to operating system

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

```
class base1
{
    public:
        int i;
        float f;
    void fun()
    {}
};
class base2
{
    public:
        int j;
        float g;
    void gfun()
    {}
};
class derived : public base1, base2
{
    public :
```

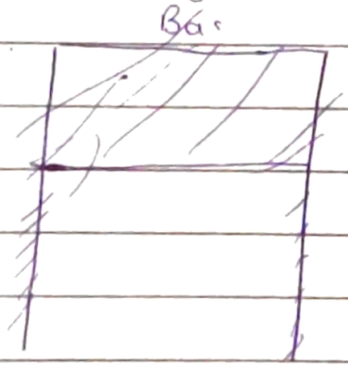
```
    int i;
    double d;
    void sum()
    {}
};
int main()
{
    derived dobj;
}
```

1] This code use multiple level Inheritance

2] In multiple level Inheritance one class can inherit from two or more classes

3]

Object layout



dobj

100		
104	i	Base1::i
108	f	Base1::f
112	j	Base2::j
116	g	Base2::g
120	i	Derived::i
124	Padding	
132	d	Derived::d

- Size of the object is 32 bytes as padding gets added in class derived.

4] Derived object dobj can access behaviours & characteristics of class Base1 & class Base 2 as well.

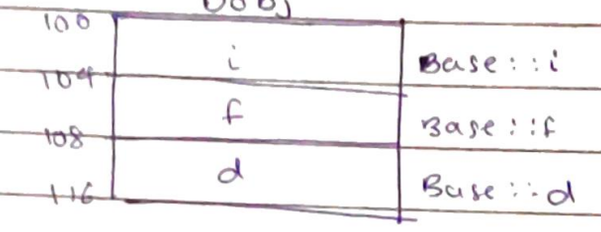
Q1. Draw object layout & class diagram of below code snippets & explain its internal working in detail. Explain the type of inheritence in the below code snippet.

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

    void sun()
    {}
};
class derivedX : public derived
{
    public
    int k;
    void run()
    {}
}
```

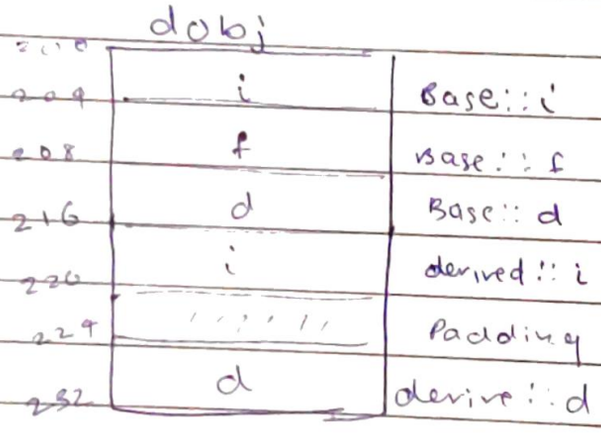
```
int main()
{
    base bobj;
    derived dobj;
    derivedX dobj1;

    return 0;
}
```

Ans Object Layout

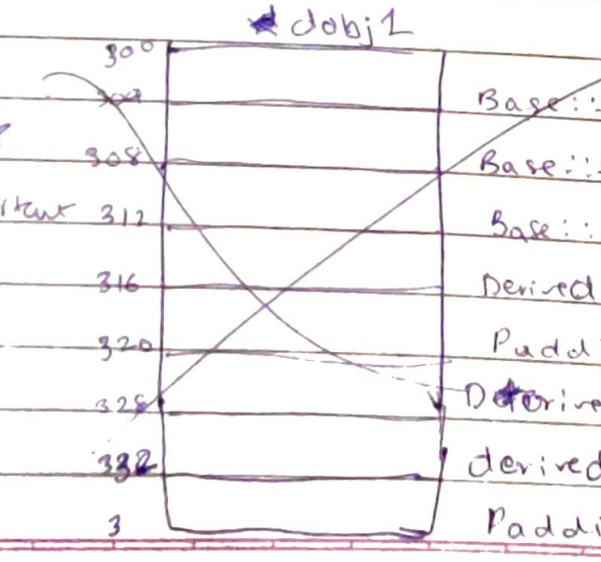
bobj



dobj

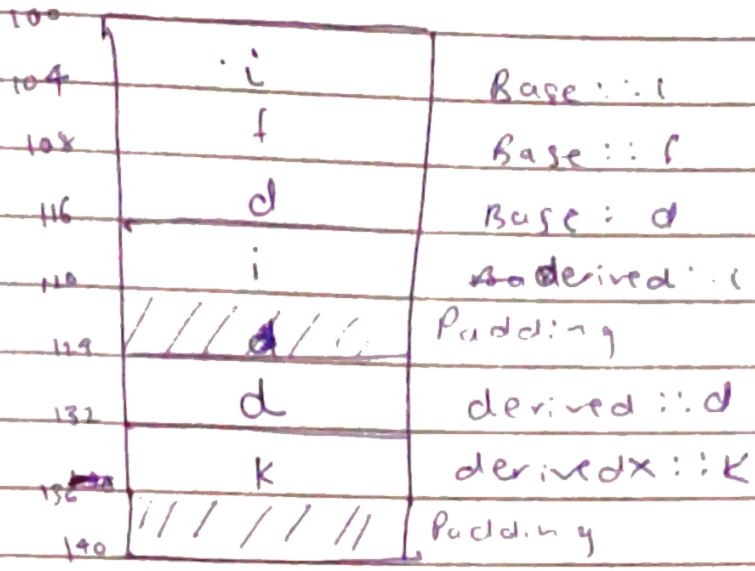


dobj1



Important

dobj1



1] The above inheritance is multi-level Inheritance

2] In case of multi-level Inheritance, there should be at least 3 classes.

3] In this case, Padding gets added derived object inherits characteristics & behaviours of Base class

4] Also the derivedX object (dobj1) inherits characteristics & behaviours of derived class indirectly also Base class.

5] Padding gets added in derived & derivedX class.

Class Diagram

	Base class	Derived class	DerivedX class
Characteristics	int i	int i	int k
	float f	double d	
	double d		
Behaviours	void fun()	void sun()	void run()
	void gun()		