

Assignment No- 20

Q1) Draw object layout of below code and explain the internal working in detail.

Class base

{ public :

int i;

float f;

double d;

void func() { }

void gunc() { }

}

class Derived : public base

{ public :

int i;

double d;

void sun() { }

void fun() { }

}

int main()

{ base bobj;

Derived dobj;

return 0;

}

→

base



Derived

bobj

i	f	d
---	---	---

} characteristics

fun()	gunc()
-------	--------

} Behaviours.

dobj

i	f	d

base::i

base::f

base::d

derived::i

derived::f

derived::d

fun()	gunc()
-------	--------

} base

sun()	func()
-------	--------

} derived.

Single level Inheritance

The derived class inherits all members of base

2) class base1

{ public:

int i;

float f;

void gun() { }

}

base1



base2



Derived

class base2

{ public:

int j;

float g;

void fun() { }

}

class Derived : public base1, base2

{ public:

int i;

double d;

void sun() { }

void fun() { }

}

int main()

{ Derived dObj;

return 0;

}

i	base1::i
f	base1::f
j	base2::j
g	base2::g
d	derived::d
gun()	base1
fun()	base2
sun()	Derived

Multilevel Inheritance

Derived class inherits both base1 and base2.

The constructor of derived will first call the constructors of base1 and base2.

Then the members of derived are initialized.

3) class base

{ public :

int i;

float f;

void func() {} // 1000

virtual void gun() {} // 2000

}

class Derived : public base

{ public :

int i;

double d;

virtual void fun() {} // 3000

void gun() {} // 4000

virtual void sun() {} // 5000

}

int main()

{ Derived dobj;

return 0;

}

→ base has a vtable containing only gun().

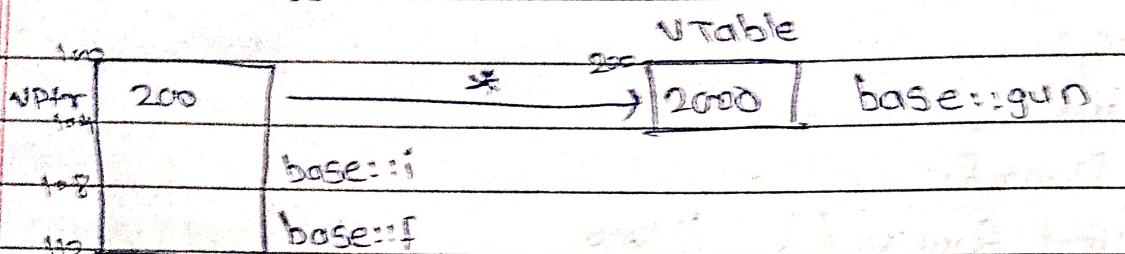
Derived create a new vtable updating the function pointers.

• When derived is instantiated -

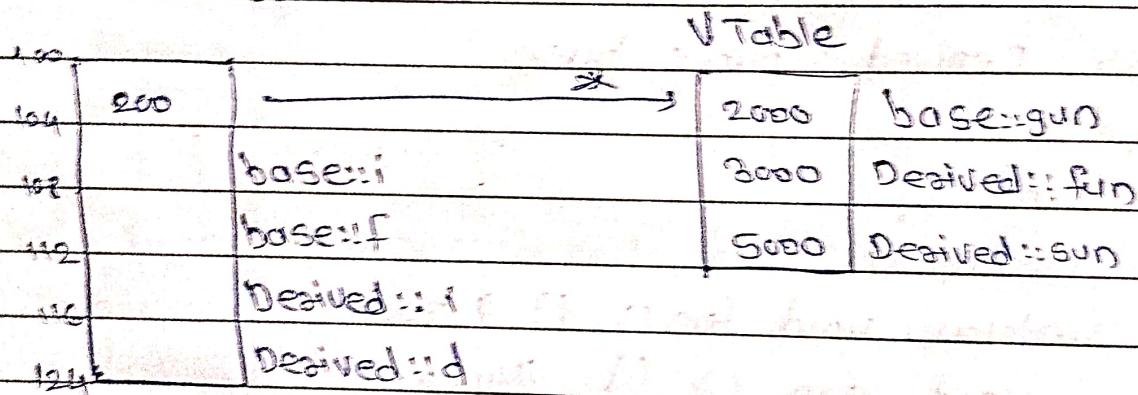
Memory is allocated for both base and derived parts

- VTable pointer (vptr) in Derived is set to point to Derived's VTable.
- Base Constructor runs first
- Derived Constructor runs afterward.

Base class



Derived class



4) class base

{ public:

int i;

float f;

void fun() { } 1000

virtual void gun() { } //2000

virtual void sun() { } //3000

void run() { } //4000

};

class Derived : public base
{ public:

int i;

double d;

virtual void fun () { } // 5000

virtual void gun () { } // 6000

void sun () { } // 7000

virtual void run () { } // 8000

};

int main ()

{ Derived dobj;

return 0;

}

Base class

100	200		2000	base::gun
104		base::i	3000	base::sun
108		base::f		
112				

Derived class

2000		2000	base::gun
	base::i	3000	base::sun
	base::f	5000	derived::fun
	derived::i	6000	derived::gun
	derived::d	8000	derived::run

working -

the constructor of base runs first.

Then, the derived constructor executes.

The VTable pointer (vptr) is updated to point to
Derived's VTable.

5] class base

{ public:

int i;

float f;

virtual void gun() {} // 1000

virtual void sun() {} // 2000

}

class Derived : public base

{ public:

int i;

double d;

virtual void fein() {} // 3000

void gun() {} // 4000

virtual void sun() {} // 5000

}

int main()

{ Derived dobj,

return 0;

}

Working -

The constructor of base runs first.
Then the derived constructor executes.
The vTable pointer is updated to point to
derived's vTable.

Base class



vTable

V PTR	200	*	1000	base::gun
:			2000	base::sun
j				

Derived class

V PTR	3200	*	1000	base::gun
base::i			2000	base::sun
derived::d			3000	derived::fun
			5000	derived::sun