Inheritance

Inheritance is a mechanism in which one class inherits the property of other class is known as Inheritance

* To drive a new class from existing class is called Inheritance.
  → new class is also known as sub class or derived class.
  → Old Class or existing class is also known as base class or super class

Basic Syntax of Inheritance.

```
class derivedclass_name : accessmode baseclass_name.
```

* Derived Class or Base Class

Derived Class:- A derived class is defined as the class derived from base class.

Syntax:
```
class derivedclass : access_mode baseclass
{
----
};
```

→ name of the derived class.
→ access mode can be public or private.
→ and name of the base class.

class:- base class is a class in OOP, from which other classes

base class is also called parent class or class.

Syntax:- class base_class_name
{
-
}

Base class

lass

2. Multiple Inheritance

1 Single Inheritance

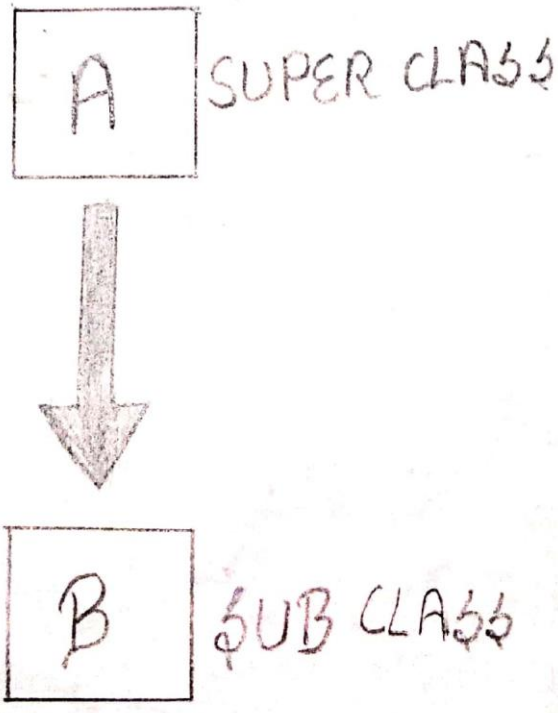
A Class which contain only one base class and only one derive class is called Single Inheritance.

It is the process of Creating a new class from an existing base class.

In Single Inheritance, a class is allowed to inherit from one class.
(One sub class is inherited by one base class).

Syntax:-

```
class derived_class_name : access_mode base_classname
{
// body of derivedclass
}
```



Example

// WAP of Single Inheritance.

\1Ł\u¢le I O1t0 h Y

LÎ4¿ț tÎ$ș,Č

țu'1f°

Volt ßt/| I\oŁ 4, t riŁ bÎ

tD\JŁ« ' ßdğiŃDț Îò ”<’S ß+b« e1ğ I ,

t’i

ćl ast ğŁCoct ț\JÏiltc -t*îbS:È

S

g'''*'¦ *

vo›ż *›*ï''°3 **

Ł

oüd t lu‹o1.

* yap\ +t«trtł

č

C|Tscr Ł1”,

seo»ś *'

getch(),

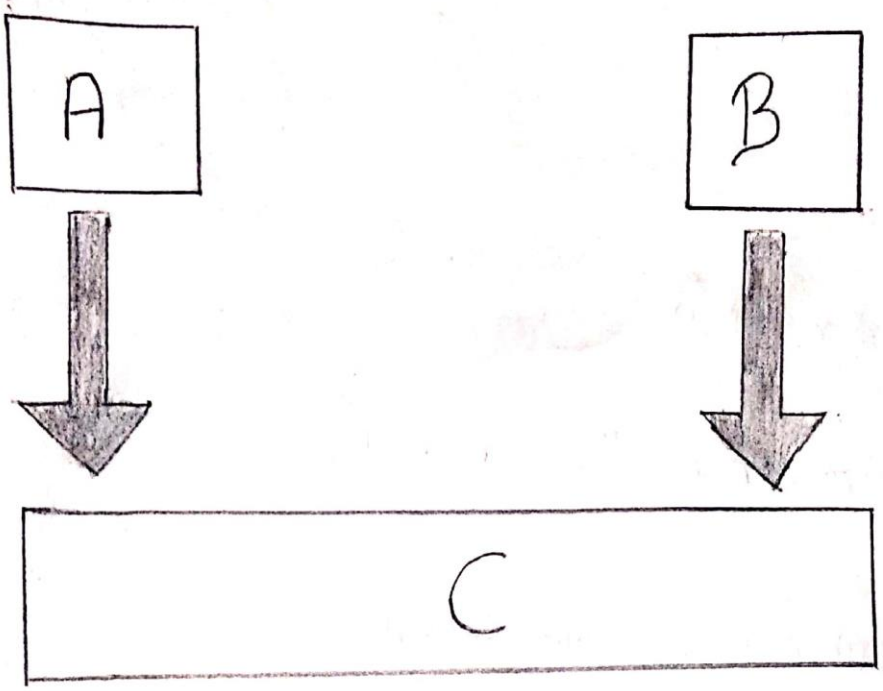
ɣ.

{Output!- addition is 30}

2. Multiple Inheritance

A class can be derived from more than one base class, then such type of Inheritance is called multiple Inheritance.

It is a process of creating a new class from more than one base class.



The General representation (Syntax)

```
Class A // base class A
{

};
Class B // base class B
{

};
Class C: access_mode A, access_mode B // derived class C which is derived from base class A and base class B
{
};
```

// Use of Multiple Inheritance

```
#include<iostream.h>
#include<conio.h>
class B1
{
 public:
void add (int a, int b)
{
```

```
public:
void sub(int x, int y)
```

```
public:
void mul(int a, int b)
```

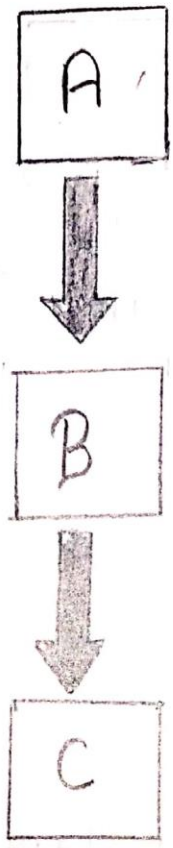
```
void main()
{
clrscr();
third t;
t.add(20,30);
```

```
t.sub(70,40);
t.mul(10,20);
getch();
}
```

(3) Multilevel Inheritance

A Process, deriving a class from another 'derived class'.

The mechanism of deriving a class from another derived class is known as Multilevel Inheritance.



Syntax:-

```
Class A
{
----
}
Class B : public A
{
----
};
Class C : Public B {
----
};
```

// WAP of Multilevel Inheritance

```
#include<iostream.h>
#include<conio.h>
class first
{ public:
void add (int a, int b)
{
cout<< "addition is :"<<a+b<< endl;
}
};
class second : public first
{
public:
void sub (int a , int b)
{
cout<< "subtraction is :"<< a-b<< endl;
}
};
class third : public second
{
public:
void mul (int a, int b)
{
cout<< "multiplication is :"<< a*b<< endl;
}
};
void main()
{
clrscr();
```

```
third t;
t.add (10,20);
t.sub (20,10);
t.mul (6,7);
getch();
}
```

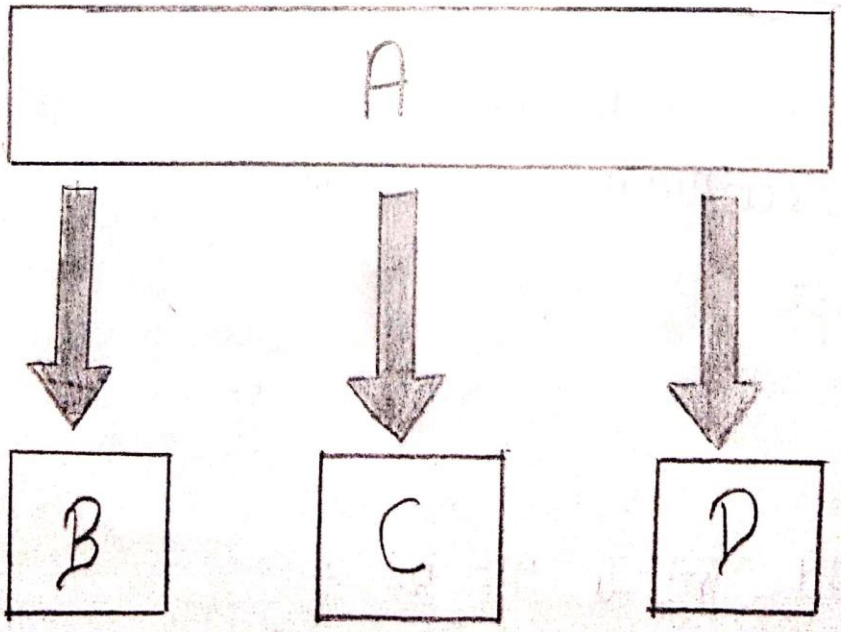
Output:- addition is 30
Subtraction is 10
Multiplication is 42

(4) Hierarchical Inheritance

In this type of Inheritance, multiple derived classes inherits from a single base class.

If more than one class is inherited from the base class, It's known as hierarchical Inheritance

for example:- Dog, Cat, Horse are derived from Animal Class.



Syntax:-

```
Class baseclass
{
    -
};
class first_derivedclass : public base_class
```

- - - -

// WAP of Hierarchical Inheritance

```
#include<iostream.h>
#include<conio.h>
```

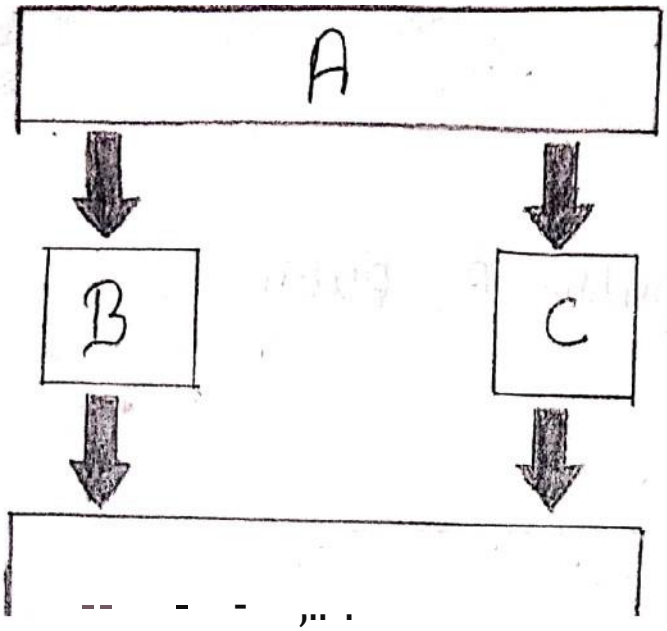
```
{
cout<< "addition is: "<< a+b<< endl;
}
};
class second: public first
{
public:
void sub (int a, int b)
{
cout<< "subtraction is : "<< a-b <<endl;
}
};
class third: public first
{
public:
void mul (int a, int b)
{
cout<< "multiplication is: "<<a*b<< endl;
}
};
class fourth: public first
{
public:
void div(int a, int b)
{
cout<< "division is : "<< a/b<< endl;
};
```

```
void main()
{
clrscr();
first f;
f.add(10,20);
second s;
s.add(3,4);
s.sub(20,10);
third t;
t.add(5,6);
t.mul(7,6);
fourth g;
g.add(1,5);
g.div(4,2);
getch();
}
```

⑤ Hybrid Inheritance:-(Virtual base Inheritance)

Hybrid Inheritance is combination of Hierarchical

It



-- - - ,... . y

In /\;;, *@' -tg , \\jQ $};yt;\ LItAs ClüSS& * ßn*

'C', and 'B', 'C' classes have a common Base class 'A'. The 'C' class inherits the features of 'A' class by two separate paths. The 'A' class is known indirect base class.

Syntax

```
class B : public A
{
 ---
};
class C : public A
{
 ---
};
class D : public B, public C
{
 ---
};
```

// WAP of Hybrid Inheritance.

```
#include<iostream.h>
#include<conio.h>
class a
{
public:
    void show()
    {
    cout<<"This is class a";
    }
};
```

```
class b : virtual public a
{
};
class c : virtual public a
{
};
class d : public b , public c
{
};
void main()
{
clrscr();
d Obj;
obj.show();
getch();
}
```

Output :- This is class a

Scanned with CamScanner