**Pure Virtual Function in C++**

**Pure Virtual Function**

A virtual function will become pure virtual function when you append "=0" at the end of declaration of virtual function. Pure virtual function doesn't have body or implementation. We must implement all pure virtual functions in derived class.

Pure virtual function is also known as abstract function.

A class with at least one pure virtual function or abstract function is called abstract class. We can't create an object of abstract class. Member functions of abstract class will be invoked by derived class object.

**Example of pure virtual function**

#include<iostream.h>

#include<conio.h>

class BaseClass **//Abstract class**

{

public:

**virtual** void Display1()**=0;** **//Pure virtual function or abstract function**

**virtual** void Display2()**=0;** **//Pure virtual function or abstract function**

void Display3()

{

cout<<"\n\tThis is Display3() method of Base Class";

}

};

class DerivedClass : public BaseClass

{

public:

void Display1()

{

cout<<"\n\tThis is Display1() method of Derived Class";

}

void Display2()

{

cout<<"\n\tThis is Display2() method of Derived Class";

}

};

void main()

{

DerivedClass D;

D.Display1(); **// This will invoke Display1() method of Derived Class**

D.Display2(); **// This will invoke Display2() method of Derived Class**

D.Display3(); **// This will invoke Display3() method of Base Class**

}

Output :

This is Display1() method of Derived Class

This is Display2() method of Derived Class

This is Display3() method of Base Class