**Lab 7: VIRTUAL FUNCTION**

**Objective:**

* To understand the concept of virtual function.
* To learn about the pure virtual function

**Theory:**

**Virtual function**

A virtual function is a member function of a class that is declared using the keyword virtual within a base class and is intended to be overridden in derived classes. The main purpose of a virtual function is to enable runtime polymorphism, allowing the program to decide at runtime which function to invoke, based on the actual type of the object being referenced, rather than the type of the pointer or references. It facilitates dynamic(late) binding which enables the correct function to be called for the object regardless of the type of references or pointer used. It supports the principle of polymorphism, a fundamental pillar of object oriented programming.

Syntax:

Class Base{

public:

virtual void functionName();

};

**Rules for using virtual function**

* It cannot be declared as static
* The function signature(prototype) of the virtual function must be identical in both the base and derived classes.
* Virtual function are accessed through pointers or references to the base class.
* Virtual function can be declared as friend function of other classes.
* Overriding a virtual function in the derived class is optional.
* Constructors cannot be virtual.

**Pointer to derived class**

a base class pointer can point to an object of its derived class. This is possible because of inheritance and allows for polymorphic behavior when using virtual function. When a base class pointer is assigned the address of a derived class object, it can only access the base clasmembers.But, if the base class contains virtual functions, and those are overridden in the derived class, the derived class versions are invoked at runtime using the base pointer.

Syntax:

Class Base{

public:

virtual void display() {

cout << "Base class function" << endl;

}};

class Derived : public Base {

public:

void display() override {

cout << "Derived class function" << endl;

}};

int main() {

Base \*ptr;

Derived obj;

ptr = &obj;

}

ptr->display();

return 0;

**Pure virtual function**

A pure virtual function is a special type of virtual function in C++ that is declared within a base class and does not have any implementation in that class. It is specified by assigning = 0 in the function declaration. Such a function serves as a placeholder, forcing derived classes to provide their own implementation. A class containing at least one pure virtual function is known as an abstract class, and it cannot be instantiated directly.

**Characteristics of pure virtual function**

* Declared using virtual keyword followed by =0
* Has no function body in the base class.
* Makes the base class an abstract class.
* Any base class that doesn’t override the pure virtual function remains abstract.

**Syntax**:

Class Base{

Virtual void functionName()=0;

};