

Date:

Roll No.:

--	--	--	--	--	--	--	--	--	--

WEEK 9

Program No:9.1

Develop a C++ program to demonstrate the use of virtual functions to achieve dynamic dispatch and enable runtime polymorphism.

Aim: Develop a C++ program to demonstrate the use of virtual functions to achieve dynamic dispatch and enable runtime polymorphism.

Description:

This C++ program demonstrates runtime polymorphism using virtual functions. It defines a base class with a virtual method and derived classes that override it. A base class pointer is used to call the overridden methods, showcasing dynamic dispatch—where the function call is resolved at runtime based on the actual object type.

Syntax:

```
class Base {
public:
    virtual void functionName(); // Virtual function
};

class Derived : public Base {
public:
    void functionName() override; // Overriding virtual function
};

int main() {
    Base* ptr;
    Derived obj;
    ptr = &obj;
    ptr->functionName(); // Runtime polymorphism via virtual function
}
```

Program:

```
#include<iostream>
using namespace std;
class shape
{
public:
    virtual void draw()
    {
        cout<<"Draw a generic shape"<<endl;
    }
};

class circle:public shape
```

Date:

Roll No.:

--	--	--	--	--	--	--	--	--	--

```
{
    public:

    void draw() override
    {
        cout<<"drawing a circle"<<endl;
    }
};

class rectangle:public
    shape
{
    public:
    void draw() override
    {
        cout<<"drawing a rectangle"<<endl;
    }
};

class triangle:public
    shape
{
    public:
    void draw() override
    {
        cout<<"drawing a triangle"<<endl;
    }
};

int main()
{
    cout<<"Roll no:24B11AI389"<<endl;
    shape* shapePtr;
    circle c;
    shapePtr=&c;
    shapePtr->draw();
    rectangle r;
    shapePtr=&r;
    shapePtr->draw();
    triangle t;
    shapePtr=&t;
    shapePtr->draw();
}
```



ADITYA
UNIVERSITY

Date:

Roll No.:

--	--	--	--	--	--	--	--	--	--

```
shape s;  
shapePtr=&s;  
shapePtr->draw();  
return 0;  
}
```

Output:

Roll no:24B11AI389

drawing a circle

drawing a rectangle

drawing a triangle

Draw a generic shape



A D I T Y A
U N I V E R S I T Y

Date:

Roll No.:

--	--	--	--	--	--	--	--	--	--

Program No :9.2

Develop a C++ program that illustrates runtime polymorphism using virtual functions.

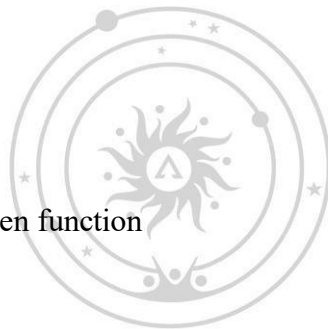
Aim : To develop a C++ program that illustrates runtime polymorphism using virtual functions

Description :

This C++ program illustrates runtime polymorphism using virtual functions. A base class declares a virtual method, and derived classes override it. A base class pointer is used to invoke the method, and due to dynamic dispatch, the correct derived class method is called at runtime, demonstrating polymorphic behavior.

Syntax :

```
class Base {  
public:  
    virtual void show(); // Virtual function  
};  
class Derived : public Base {  
public:  
    void show() override; // Overridden function  
};  
int main() {  
    Base* ptr;  
    Derived obj;  
    ptr = &obj;  
    ptr->show(); // Calls Derived's show() at runtime  
}
```



ADITYA
UNIVERSITY

Program:

```
#include <iostream>  
using namespace std;  
class Animal {  
public:  
    virtual void makeSound() {  
        cout << "Animal makes a sound" << endl;  
    }  
};
```

Date:

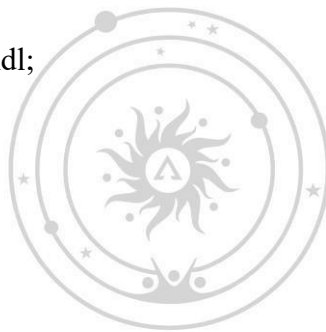
Roll No.:

--	--	--	--	--	--	--	--	--	--

```
class Dog : public Animal {  
public:  
    void makeSound() override {  
        cout << "Dog barks" << endl;  
    }  
};
```

```
class Cat : public Animal {  
public:  
    void makeSound() override {  
        cout << "Cat meows" << endl;  
    }  
};
```

```
int main() {  
  
    cout<<"Roll no:24B11AI389"<<endl;  
    Animal* animalPtr;  
    Dog d;  
    Cat c;  
    animalPtr = &d;  
    animalPtr->makeSound();  
    animalPtr = &c;  
    animalPtr->makeSound();  
    return 0;  
}
```



A D I T Y A
U N I V E R S I T Y

Output:

```
Roll no:24B11AI389  
Dog barks  
Cat meows
```