

**OOP (Object Oriented Programming) Lab**

**LAB REPORT # 8**

**Semester**: 2ndSemester

**Section**: C

**Submitted To:**

**Mr. Muhammad Husnain**

**Submitted By:**

**Name**: Muhammad Afzal

**Roll No**: 22-CS-035

**Code:**

// without virtual Function

#include <iostream>

using namespace std;

class Add

{

    int x = 5, y = 20;

public:

    void display() // overridden function

    {

        cout << "Value of x is : " << x + y << endl;

    }

};

class Subtract : public Add

{

    int y = 10, z = 30;

public:

    void display() // overridden function

    {

        cout << "Value of y is : " << y - z << endl;

    }

};

int main()

{

    Add \*m;      // base class pointer .it can only access the base class members

    Subtract s; // making object of derived class

    m = &s;

    m->display(); // Accessing the function by using base class pointer

    return 0;

}

**Output:**

****

**Code:**

#include <iostream>

using namespace std;

class Add

{

public:

    virtual void print()

    {

        int a = 20, b = 30;

        cout << " base class Action is:" << a + b << endl;

    }

    void show()

    {

        cout << "show base class" << endl;

    }

};

class Sub : public Add

{

public:

    void print() // print () is already virtual function in derived class, we could also declared as virtual void print() explicitly

    {

        int x = 20, y = 10;

        cout << " derived class Action:" << x - y << endl;

    }

    void show()

    {

        cout << "show derived class" << endl;

    }

};

// main function

int main()

{

    Add \*aptr;

    Sub s;

    aptr = &s;

    // virtual function, binded at runtime (Runtime polymorphism)

    aptr->print();

    // Non-virtual function, binded at compile time

    aptr->show();

    return 0;

}

**Output:**

****

**Code:**

#include <iostream>

using namespace std;

class Animal

{

public:

    virtual void show() = 0; // Pure virtual function declaration.

};

class Man : public Animal

{

public:

    void show()

    {

        cout << "Man is the part of animal husbandry " << endl;

    }

};

int main()

{

    Animal \*aptr; // Base class pointer

    // Animal a;

    Man m; // derived class object creation.

    aptr = &m;

    aptr->show();

    return 0;

}

**Output:**

****

**Code:**

#include <iostream>

using namespace std;

class Add

{

protected:

    int x = 5, y = 20;

public:

    virtual void display(){

        cout << "Value of x is: " << x + y << endl;

    }

};

class Subtract : public Add{

private:

    int y = 10, z = 30;

public:

    void display(){

        cout << "Value of y is: " << y - z << endl;

    }

};

class Animal{

public:

    virtual void show() = 0; // Pure virtual function declaration.

};

class Man : public Animal{

public:

    void show(){

        cout << "Man is a part of animal husbandry." << endl;

    }

};

int main(){

    Add addObj;

    cout << "Code 1 Output:" << endl;

    addObj.display();

    Animal \*animalPtr;

    Man manObj;

    animalPtr = &manObj;

    cout << "\nCode 3 Output:" << endl;

    animalPtr->show();

    cout << "\nCode 2 Output:" << endl;

    Add \*addPtr;

    Subtract subtractObj;

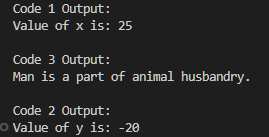
    addPtr = &subtractObj;

    addPtr->display();

    return 0;

}

**Output:**

****