

**OBJECT ORIENTED PROGRAMMING**

**LAB 4**

* Name :HITU RAJ
* Roll no. :2005025
* Branch :CSE

//Q1.WAP to swap private data\_025 member of two different classes.

//[The classes have no relation with each other].

#include <iostream>

using namespace std;

class xyz;

class abc

{

    int data\_025;

    friend void swap(abc &, xyz &);

public:

    void input()

    {

        cin >> data\_025;

    }

    void display()

    {

        cout << data\_025 << " ";

    }

};

class xyz

{

    int data\_025;

    friend void swap(abc &, xyz &);

public:

    void input()

    {

        cin >> data\_025;

    }

    void display()

    {

        cout << data\_025 << " ";

    }

};

void swap(abc &a\_025, xyz &x)

{

    int temp = a\_025.data\_025;

    a\_025.data\_025 = x.data\_025;

    x.data\_025 = temp;

}

int main()

{

    abc a\_025;

    xyz b\_025;

    cout << "enter value in 2 class \n";

    a\_025.input();

    b\_025.input();

    cout << "before swap th e values of class1 and class  is 2 \n";

    a\_025.display();

    b\_025.display();

    swap(a\_025, b\_025);

    cout << "\nAfter swap the values of class1 and class 2 is \n";

    a\_025.display();

    b\_025.display();

    return 0;

}

**OUTPUT-Q1**

//Q2.Create two classes which stores distance in feet\_025, inches\_025 and meter,

//centimeter format respectively. Write a function which compares distance

//in object of these classes and displays the larger one.

#include <bits/stdc++.h>

using namespace std;

class db\_025;

class dm\_025

{

    int m\_025;

    int cm\_025;

public:

    void input()

    {

        cin >> m\_025 >> cm\_025;

    }

    friend void compare(db\_025, dm\_025);

};

class db\_025

{

    int feet\_025;

    int inches\_025;

public:

    void input()

    {

        cin >> feet\_025 >> inches\_025;

    }

    friend void compare(db\_025, dm\_025);

};

void compare(db\_025 a, dm\_025 b)

{

    int tm, tfim;

    tm = b.m\_025 + b.cm\_025 / 100;

    tfim = (a.feet\_025 \* 0.3) + ((a.inches\_025 / 12) \* 0.3);

    if (tfim > tm)

        cout << "\nthe larget one is feet one and the value is " << a.feet\_025 << " feet " << a.inches\_025 << "inches"

             << "which is equal to " << tfim<<" metre";

    else

        cout << "\nthe larget one is meter one and the value is " << b.m\_025 << "meter" << b.cm\_025 << "cm\_025";

}

int main()

{

    db\_025 a;

    dm\_025 b;

    cout << "\nenter the value of m and cm ";

    b.input();

    cout << "\nenter the value of feet and inches ";

    a.input();

    compare(a, b);

    return 0;

}

**OUTPUT-Q2**



/\* Q3.Create a class with an integer data member. Include functions for

input and output in\_025 class. Count the number of times each function is

called and display it.\*/

#include <iostream>

using namespace std;

class inout

{

    int data\_025;

public:

    static int in\_025;

    static int out\_025;

    void input()

    {

        in\_025++;

    }

    void output()

    {

        out\_025++;

    }

    static void count()

    {

        cout << "the no.of time input and output fun called is ";

        cout << in\_025 <<" "<< out\_025;

    }

};

int inout::in\_025;

int inout::out\_025;

int main()

{

    class inout a,b;

    a.input();

    a.output();

    b.input();

    a.output();

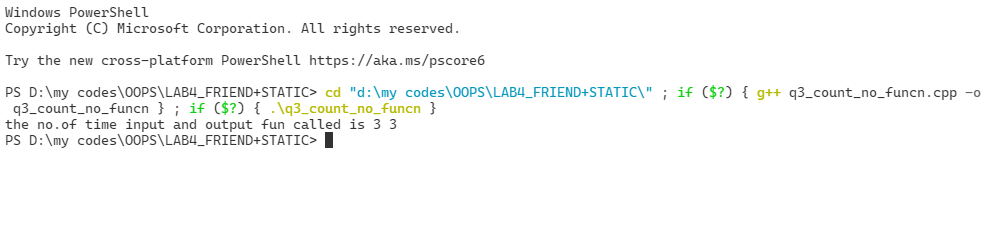
    b.input();

    a.output();

    inout::count();

    return 0;

}

 **OUTPUT-Q3**

/\* Q4.Create a class which stores name\_025, roll\_025 number and oop marks\_025 for a

student. Input data for n students. Find the average marks\_025 scored by

n students, store it as a data member of the class and display it using

a function which may be called without object.\*/

#include <bits/stdc++.h>

using namespace std;

class stud

{

    char name\_025[10];

    int roll\_025, marks\_025;

    static float avg;

public:

    void input()

    {

        cin >> name\_025 >> roll\_025 >> marks\_025;

    }

    static void avrg(stud a[], int n)

    {

        int sum = 0;

        for (int i = 0; i < n; i++)

        {

            sum = sum + a[i].marks\_025;

        }

        avg = sum / n;

        cout << "average of the data is" << avg;

    }

};

float stud::avg;

int main()

{

    stud a[5];

    int n;

    cout << "how many students u have";

    cin >> n;

    cout<<"enter the name roll and oop marks\n";

    for (int i = 0; i < n; i++)

    {

        a[i].input();

    }

    stud::avrg(a, n);

    return 0;

**OUTPUT-Q4**

}



/\*Q5.Create a class which stores name\_025, author\_025 and price\_025 of a book. Store

information for n number of books. Display information of all the

books in a given price\_025 range using friend function in a tabular

format.\*/

#include <bits/stdc++.h>

using namespace std;

class book

{

    char name\_025[10], author\_025[10];

    int price\_025;

public:

    void input()

    {

        cin >> name\_025 >> author\_025 >> price\_025;

    }

    friend void range(book a[], int n, int upl\_025, int lowl\_025);

};

void range(book a[], int n, int upl\_025, int lowl\_025)

{

    for (int i = 0; i < n; i++)

    {

        if (a[i].price\_025 >= lowl\_025 && a[i].price\_025 <= upl\_025)

        {

            cout << a[i].name\_025 << "\t" << a[i].author\_025 << "\t" << a[i].price\_025;

        }

        cout << "\n";

    }

}

int main()

{

    int n, upl\_025, lowl\_025;

    book a[4];

    cout << "how many books u have";

    cin >> n;

    cout << "\nenter the name\_025 author\_025 and price\_025 of each book ";

    for (int i = 0; i < n; i++)

    {

        a[i].input();

    }

    cout << "enter lower limit and enter the upper one";

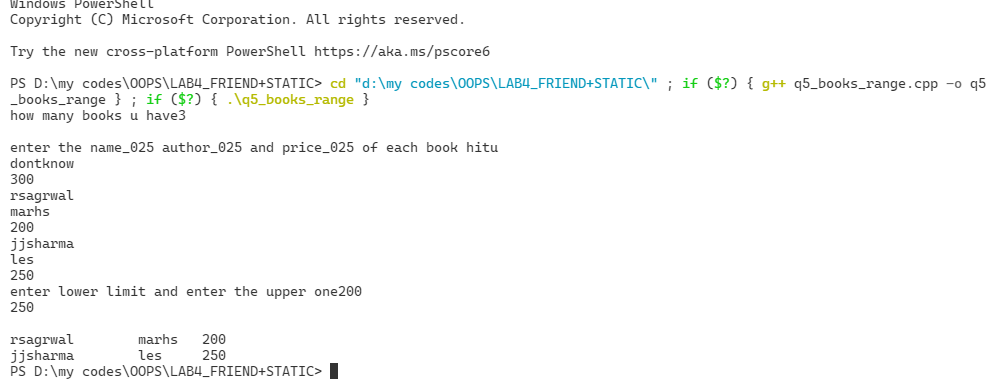
    cin >> lowl\_025 >> upl\_025;

    range(a, n, upl\_025, lowl\_025);

    return 0;

}

**OUTPUT-Q5**



/\*Q6Write a program to find out the greatest between two numbers

defined in two different classes by using friend function.

[using friend of one and member of the other class]\*/

//#include <bits/stdc++.h>

#include <iostream>

using namespace std;

class b\_025;

class a\_025

{

    int no1\_025;

public:

    void input()

    {

        cin >> no1\_025;

    }

    void greatest(b\_025);

};

class b\_025

{

    int no2\_025;

public:

    void input()

    {

        cin >> no2\_025;

    }

    friend void a\_025::greatest(b\_025);

};

void a\_025::greatest(b\_025 obj)

{

    if (no1\_025 > obj.no2\_025)

        cout << "greatest no. is " << no1\_025;

    else

        cout << "greatest no. is " << obj.no2\_025;

}

int main()

{

    a\_025 obj1;

    b\_025 obj2;

    cout << "enter the value in 2 different class";

    obj1.input();

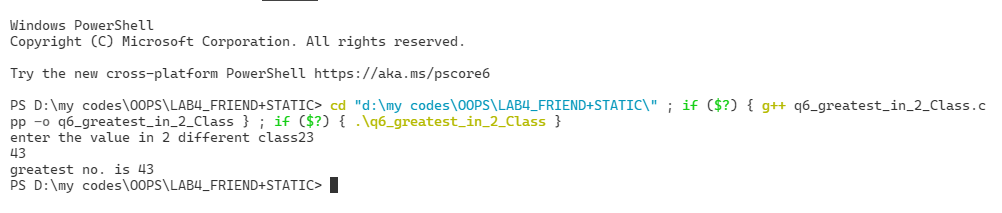
    obj2.input();

    obj1.greatest(obj2);

    return 0;

};

**OUTPUT-Q6**



/\*Q7.Create two classes DM and DB which store the value of distances.

DM stores distances in meters and centimeters and DB in feet\_025 and

inches\_025. Write a program that can read values for the class objects and

add one object of Dm with another object of DB. Use friend function

to carry out the operation. The object that stores the results may be a

DM object or DB object, depending on the units in which the results

are required. The display should in the format of feet\_025 and inches\_025 or

meters and centimeters depending on the object on display.\*/

#include <bits/stdc++.h>

using namespace std;

class db;

class dm

{

    int m\_025;

    int cm\_025;

public:

    void input()

    {

        cin >> m\_025 >> cm\_025;

    }

    friend void add(db, dm);

};

class db

{

    int feet\_025;

    int inches\_025;

public:

    void input()

    {

        cin >> feet\_025 >> inches\_025;

    }

    friend void add(db, dm);

};

void add(db a, dm b)

{

    b.m\_025 = b.m\_025 + 0.3 \* a.feet\_025;

    b.cm\_025 = b.cm\_025 + (0.3 / 12 \* a.inches\_025);

    cout << "\nadded value of feet and meter is " << b.m\_025 << "m " << b.cm\_025 << "cm ";

}

int main()

{

    db a;

    dm b;

    cout << "\nenter the value of m and cm ";

    b.input();

    cout << "\nenter the value of feet and inches";

    a.input();

    add(a, b);

    return 0;

}

**OUTPUT-Q7**

