**Assignment 4 (OOPs)**

**Name: Yatin Kanwar   
Branch: ECE  
SID: 23105057**

**Q1. To find highest of two given numbers in two different classes using friend function.**

#include <iostream>

using namespace std;

class ClassB;

class ClassA {

    int num1;

public:

    ClassA(int *n*) : num1(n) {}

    friend void findHighest(ClassA, ClassB);

};

class ClassB {

    int num2;

public:

    ClassB(int *n*) : num2(n) {}

    friend void findHighest(ClassA, ClassB);

};

void findHighest(ClassA *a*, ClassB *b*) {

    if (a.num1 > b.num2)

        cout << "Highest number is: " << a.num1 << endl;

    else

        cout << "Highest number is: " << b.num2 << endl;

}

int main() {

    ClassA a(10);

    ClassB b(20);

    findHighest(a, b);

    return 0;

}

**Output:  
**

**Q2. To demonstrate the use of friend function with inline assignment.**

#include <iostream>

using namespace std;

class B;

class A {

    int x;

public:

    A(int *val*) : x(val) {}

    friend void showValue(A, B);

};

class B {

    int y;

public:

    B(int *val*) : y(val) {}

    friend void showValue(A, B);

};

void showValue(A *a*, B *b*) {

    cout << "Value in A: " << a.x << endl;

    cout << "Value in B: " << b.y << endl;

}

int main() {

    A a(10);

    B b(20);

    showValue(a, b);

    return 0;

}

**Output:**

****

**Q3. To enter any number and find its factorial and display it using constructor.**

#include <iostream>

using namespace std;

class Factorial {

    long long fact;

public:

    Factorial(int *n*);

    void display();

};

Factorial::Factorial(int n) {

    fact = 1;

    for (int i = 1; i <= n; ++i) {

        fact \*= i;

    }

}

void Factorial::display() {

    cout << "Factorial: " << fact << endl;

}

int main() {

    int num;

    cout << "Enter a number: ";

    cin >> num;

    Factorial f(num);

    f.display();

    return 0;

}

**Output:**

****

**Q4. to enter two operands using constructors and perform arithmetic operations(using +,-,\*,/) on them and display the results.**

#include <iostream>

using namespace std;

class Arithmetic {

    float a, b;

public:

    Arithmetic(float *x*, float *y*) : a(x), b(y) {}

    void add() { cout << "Sum: " << a + b << endl; }

    void subtract() { cout << "Difference: " << a - b << endl; }

    void multiply() { cout << "Product: " << a \* b << endl; }

    void divide() {

        if (b != 0)

            cout << "Quotient: " << a / b << endl;

        else

            cout << "Division by zero error" << endl;

    }

};

int main() {

    float num1, num2;

    cout << "Enter first operand: ";

    cin >> num1;

    cout << "Enter second operand: ";

    cin >> num2;

    Arithmetic ar(num1, num2);

    ar.add();

    ar.subtract();

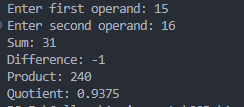
    ar.multiply();

    ar.divide();

    return 0;

}

**Output:**



**Q5. Imagine a ticket selling booth at a fair. People passing by are requested to purchase a ticket. A ticket is priced as Rs.2.50/ the booth keeps track of the number of people that have visited the booth and ,of the total amount of money collected. Model this ticket selling booth with a class called ticbooth including following members.**

**Data Members :  
Number of people visited  
Total amount of money collected**

**Member functions:  
-To assign initial values(assign 0 to both data members)  
- To increment people total only in case ticket is not sold out  
-To increment people total as well as amount total if a ticket is sold out  
-To display the two totals  
-To display the number of ticket sold out (a ticket one)  
Include a program to test this class.**

#include <iostream>

using namespace std;

class TicBooth{

    int peopleVisited;

    float totalMoney;

    public:

    TicBooth(): peopleVisited(0), totalMoney(0.0) {}

    void incrementPeopleOnly(){

        peopleVisited++;

    }

    void sellTicket(){

        peopleVisited++;

        totalMoney+=2.50;

    }

    void displayTotals(){

        cout << "Total people visited: " << peopleVisited << endl;

        cout << "Total money collected: Rs. " << totalMoney << endl;

    }

    void displayTicketsSold() {

        cout << "Total tickets sold: " << peopleVisited << endl;

    }

};

int main(){

    TicBooth booth;

    booth.incrementPeopleOnly();

    booth.sellTicket();

    booth.sellTicket();

    booth.displayTotals();

    booth.displayTicketsSold();

    return 0;

}

**Output:**

