

**International University of Business Agriculture and Technology**

**Lab Report 10**

**Course Code: CSC 284**

**Course Name: Programming in C++ Lab**

**Submitted To:**

**Engr. A.S.M. Shakil Ahamed**

Senior Lecturer

Dept. of Computer Science and Engineering

International University of Business Agriculture and Technology

**Submitted By:**

Name: Md. Mahfujar Rahman

ID: 23303151

Section: C

**1.Code:**

#include <iostream>

using namespace std;

class Cart

{

private:

    double price;

    int quantity;

public:

    Cart(double p, int q) : price(p), quantity(q) {}

    double getTotal() const { return price \* quantity; }

    double operator+(const Cart &other)

    {

        return this->getTotal() + other.getTotal();

    }

};

int main()

{

    Cart cart1(10.0, 5);

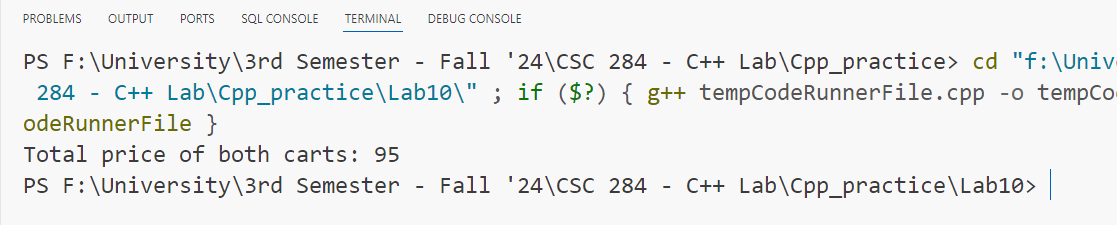
    Cart cart2(15.0, 3);

    cout << "Total price of both carts: " << cart1 + cart2 << endl;

    return 0;

}

**Output:**

****

**2.Code:**

#include <iostream>

using namespace std;

class Distance

{

private:

    int kilometers;

    int meters;

public:

    Distance(int km, int m) : kilometers(km), meters(m) {}

    void display() const

    {

        cout << kilometers << " km " << meters << " m" << endl;

    }

    Distance operator+(const Distance &other)

    {

        int totalMeters = meters + other.meters;

        int totalKilometers = kilometers + other.kilometers + totalMeters / 1000;

        totalMeters %= 1000;

        return Distance(totalKilometers, totalMeters);

    }

};

int main()

{

    Distance distance1(5, 700);

    Distance distance2(3, 800);

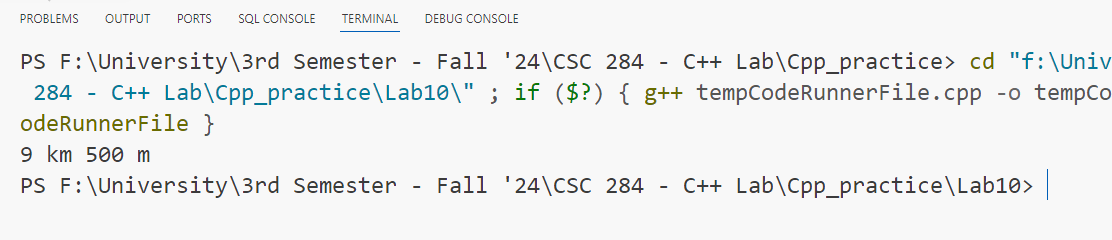
    Distance total = distance1 + distance2;

    total.display();

    return 0;

}

**Output:**

****

**3.Code:** #include <iostream>

using namespace std;

class Account

{

private:

    double balance;

public:

    Account(double b) : balance(b) {}

    double getBalance() const { return balance; }

    Account &operator-(double amount)

    {

        balance -= amount;

        return \*this;

    }

};

int main()

{

    Account acc(1000.0);

    cout << "Initial balance: " << acc.getBalance() << endl;

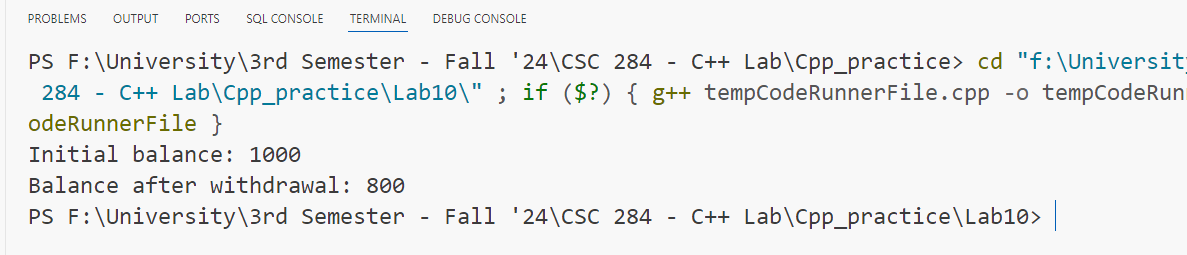
    acc - 200.0;

    cout << "Balance after withdrawal: " << acc.getBalance() << endl;

    return 0;

}

**Output:**



**4.Code:**

#include <iostream>

using namespace std;

class Complex

{

private:

    double real;

    double imag;

public:

    Complex(double r, double i) : real(r), imag(i) {}

    void display() const

    {

        cout << real << " + " << imag << "i" << endl;

    }

    Complex operator+(const Complex &other)

    {

        return Complex(real + other.real, imag + other.imag);

    }

    Complex operator-(const Complex &other)

    {

        return Complex(real - other.real, imag - other.imag);

    }

};

int main()

{

    Complex num1(3, 4);

    Complex num2(1, 2);

    Complex sum = num1 + num2;

    Complex diff = num1 - num2;

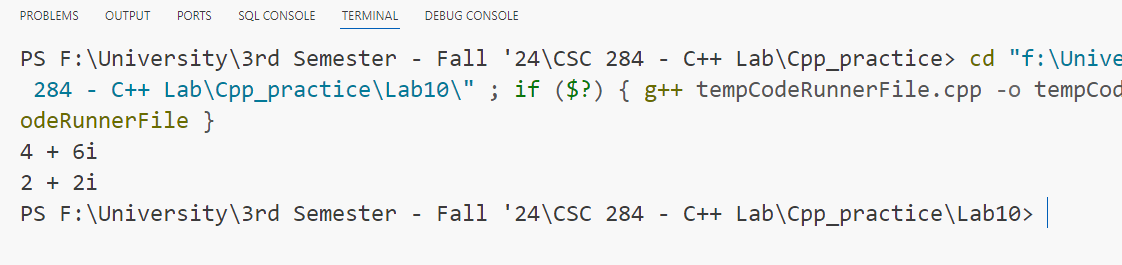
    sum.display();

    diff.display();

    return 0;

}

**Output:**



**5.Code:**

#include <iostream>

using namespace std;

class Matrix{

private:

    int mat[2][2];

public:

    Matrix(int a, int b, int c, int d){

        mat[0][0] = a;

        mat[0][1] = b;

        mat[1][0] = c;

        mat[1][1] = d;

    }

    void display() const{

        for (int i = 0; i < 2; ++i)

        {

            for (int j = 0; j < 2; ++j)

            {

                cout << mat[i][j] << " ";

            }

            cout << endl;

        }

    }

    Matrix operator+(const Matrix &other)

    {

        return Matrix(mat[0][0] + other.mat[0][0], mat[0][1] + other.mat[0][1],

                      mat[1][0] + other.mat[1][0], mat[1][1] + other.mat[1][1]);

    }

};

int main(){

    Matrix m1(1, 2, 3, 4);

    Matrix m2(5, 6, 7, 8);

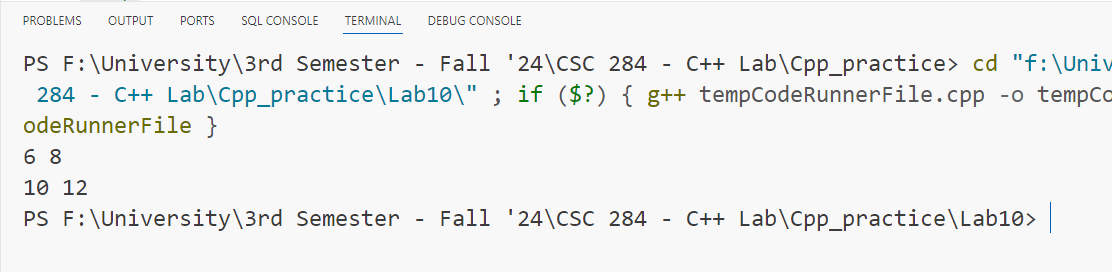
    Matrix result = m1 + m2;

    result.display();

    return 0;

}

**Output:**

****

**6.Code:**

#include <iostream>

using namespace std;

class Employee

{

private:

    string name;

    double salary;

public:

    Employee(string n, double s) : name(n), salary(s) {}

    double getSalary() const { return salary; }

    Employee operator+(double bonus)

    {

        return Employee(name, salary + bonus);

    }

};

int main()

{

    Employee emp("John", 5000);

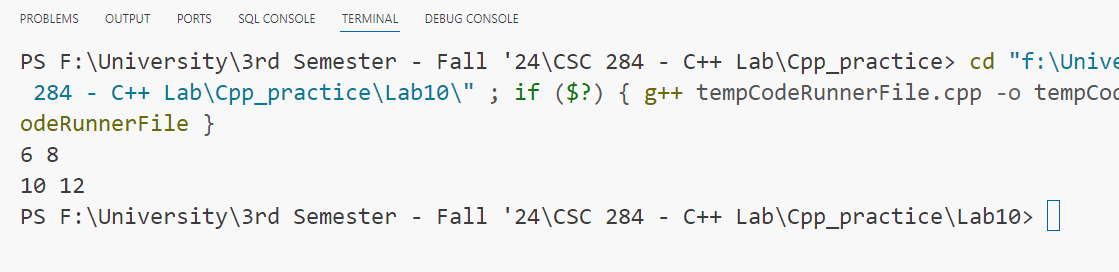
    Employee empWithBonus = emp + 1000;

    cout << "Salary after bonus: " << empWithBonus.getSalary() << endl;

    return 0;

}

**Output:**

****

**7.Code:**

#include <iostream>

using namespace std;

class Fraction

{

private:

    int numerator, denominator;

public:

    Fraction(int num, int denom) : numerator(num), denominator(denom) {}

    void display() const

    {

        cout << numerator << "/" << denominator << endl;

    }

    Fraction operator+(const Fraction &other)

    {

        int commonDenominator = denominator \* other.denominator;

        int newNumerator = numerator \* other.denominator + other.numerator \* denominator;

        return Fraction(newNumerator, commonDenominator);

    }

};

int main()

{

    Fraction frac1(1, 2);

    Fraction frac2(1, 3);

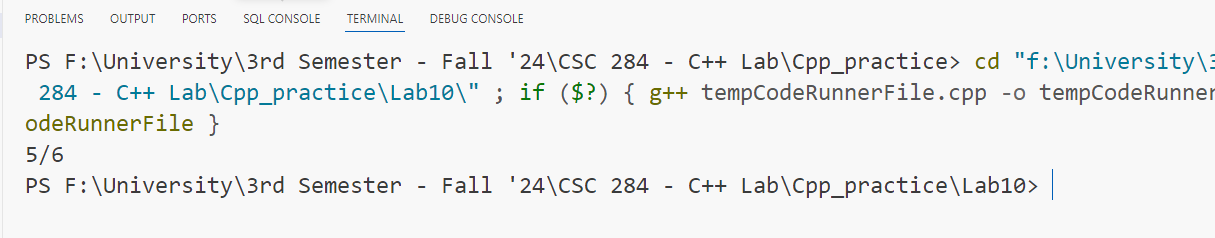
    Fraction result = frac1 + frac2;

    result.display();

    return 0;

}

**Output:**

****

**8.Code:**

#include <iostream>

using namespace std;

class Player

{

private:

    string name;

    int points;

public:

    Player(string n, int p) : name(n), points(p) {}

    int getPoints() const { return points; }

    Player operator+(const Player &other)

    {

        return Player(name + " & " + other.name, points + other.points);

    }

};

int main()

{

    Player player1("Alice", 100);

    Player player2("Bob", 150);

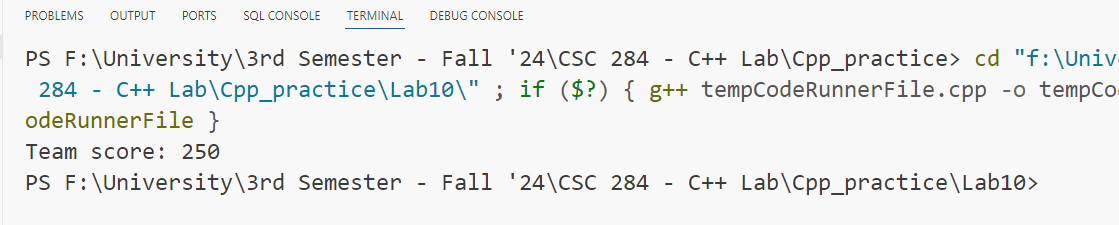
    Player team = player1 + player2;

    cout << "Team score: " << team.getPoints() << endl;

    return 0;

}

**Output:**

****

**9.Code:**

#include <iostream>

using namespace std;

class Box

{

private:

    double length, width, height;

public:

    Box(double l, double w, double h) : length(l), width(w), height(h) {}

    double getVolume() const

    {

        return length \* width \* height;

    }

    bool operator>(const Box &other)

    {

        return getVolume() > other.getVolume();

    }

};

int main()

{

    Box box1(3.0, 4.0, 5.0);

    Box box2(2.0, 6.0, 6.0);

    if (box1 > box2)

    {

        cout << "Box 1 is larger" << endl;

    }

    else

    {

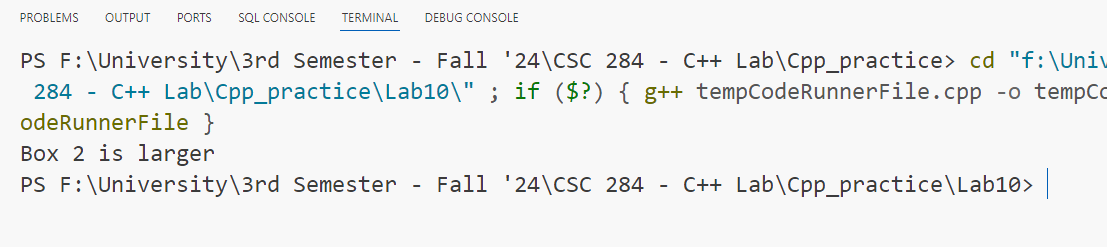
        cout << "Box 2 is larger" << endl;

    }

    return 0;

}

**Output:**

****

**10.Code:**

#include <iostream>

using namespace std;

class MyString

{

private:

    string str;

public:

    MyString(string s) : str(s)

    {

    }

    string getStr() const

    {

        return str;

    }

    MyString operator+(const MyString &other)

    {

        return MyString(str + " " + other.str);

    }

};

int main()

{

    MyString firstName("John");

    MyString lastName("Doe");

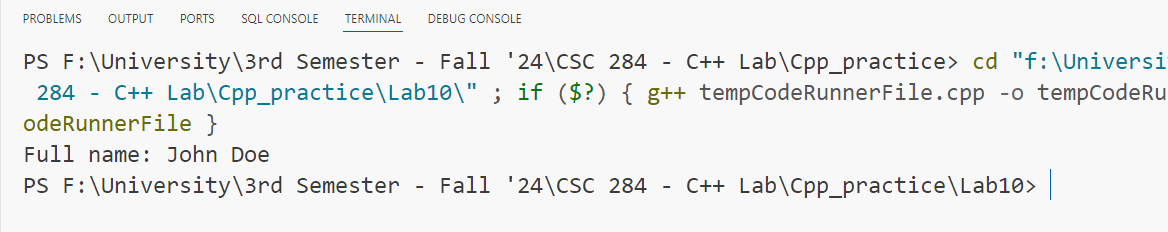
    MyString fullName = firstName + lastName;

    cout << "Full name: " << fullName.getStr() << endl;

    return 0;

}

**Output:**

****