**Task1**

#include <iostream>

using namespace std;

class Humidity;

class Temperature {

    private:

        float temperature;

    public:

        Temperature(float temp) : temperature(temp) {}

        friend float calculateHeatIndex(Temperature temp, Humidity hum);

};

class Humidity {

    private:

        float humidity;

    public:

        Humidity(float h) : humidity(h) {}

        friend float calculateHeatIndex(Temperature temp, Humidity hum);

};

float calculateHeatIndex(Temperature temp, Humidity hum) {

    return temp.temperature + 0.33 \* hum.humidity - 4;

}

int main() {

    Temperature t(31);

    Humidity h(90);

    cout << "Heat Index: " << calculateHeatIndex(t,h);

}

A computer screen with white text

AI-generated content may be incorrect.

**Task2**

#include <iostream>

using namespace std;

class Book {

    private:

        string title;

        float price;

    public:

        Book(string t, float p) : title(t), price(p) {}

        friend class Librarian;

};

class Librarian {

    public:

        void displayBookDetails(Book& b) {

            cout << "Displaying book details..." << endl << "Title: " << b.title << ", Price: " << b.price << endl;

        }

        void applyBookDiscount(Book& b, float p) {

            if(p <= 100 && p > 0) {

                cout << "Applying "  << p << "% discount on Book " << b.title << "..." << endl;

                b.price = b.price \* (100-p)/100;

            } else {

                cout << "Discount " << p << "% out of range!" << endl;

            }

        }

};

int main() {

    Book b("BFG", 71.2);

    Librarian taha;

    taha.displayBookDetails(b);

    taha.applyBookDiscount(b, 25);

    taha.displayBookDetails(b);

}

A computer screen shot of a black screen

AI-generated content may be incorrect.

**Task3**

#include <iostream>

using namespace std;

class Student {

    private:

        string name;

        char grades[3];

    public:

        Student(string n, char g1, char g2, char g3) : name(n) {

            grades[0] = g1;

            grades[1] = g2;

            grades[2] = g3;

        }

        friend float computeAverageGrade(Student& s);

        friend class Teacher;

};

class Teacher {

    public:

        void viewStudentGrade(Student s) {

            cout << "Displaying Student " << s.name << "'s grades.." << endl;

            for (int i = 0; i < 3; i++) {

                cout << "Subject " << i + 1 << ": " << s.grades[i] << endl;

            }

        }

        void updateStudentGrade(Student& s, char g1, char g2, char g3) {

            cout << "Updating Student " << s.name << "'s grades.." << endl;

            s.grades[0] = g1;

            s.grades[1] = g2;

            s.grades[2] = g3;

        }

};

float computeAverageGrade(Student& s) {

    float total = 0.0;

    for (int i = 0; i < 3; i++) {

        switch (s.grades[i]) {

            case 'A': total += 4.0; break;

            case 'B': total += 3.0; break;

            case 'C': total += 2.0; break;

            case 'D': total += 1.0; break;

            case 'F': total += 0.0; break;

            default: cout << "Invalid grade: " << s.grades[i] << endl;

        }

    }

    return total / 3.0;

}

int main() {

    Student s1("Mohammad Taha Anwer", 'A', 'A', 'A');

    Teacher t;

    t.viewStudentGrade(s1);

    cout << "Average Grade Point: " << computeAverageGrade(s1) << endl;

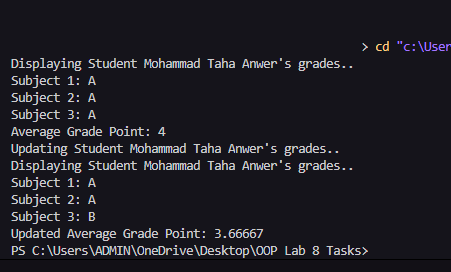
    t.updateStudentGrade(s1, 'A', 'A', 'B');

    t.viewStudentGrade(s1);

    cout << "Updated Average Grade Point: " << computeAverageGrade(s1) << endl;

    return 0;

}



**Task4**

#include <iostream>

using namespace std;

class Currency {

    private:

        float money;

    public:

        Currency(float m) : money(m) {}

        Currency operator-() {

            return Currency(-money);

        }

        Currency operator+(Currency c) {

            return Currency(money + c.money);

        }

        Currency operator-(Currency c) {

            return Currency(money - c.money);

        }

        Currency operator+=(Currency c) {

            money += c.money;

            return \*this;

        }

        Currency operator-=(Currency c) {

            money -= c.money;

            return \*this;

        }

        void display() {

            cout << "Money: " << money << endl;

        }

};

int main() {

    Currency c1(100);

    Currency c2(50);

    Currency c3 = -c2;

    Currency c4 = c1 + c3;

    Currency c5 = c3 - c2;

    c1 += c2;

    c2 -= c3;

    c1.display();

    c2.display();

    c3.display();

    c4.display();

    c5.display();

}

A computer screen shot of a computer

AI-generated content may be incorrect.

**Task5**

#include <iostream>

using namespace std;

class Fraction {

    private:

        int numerator;

        int denominator;

    public:

        Fraction(int num = 0, int den = 1) {

            if (den == 0) {

                cout << "Denominator can't be zero! Set as 1" << endl;

                den = 1;

            }

            numerator = num;

            denominator = den;

        }

        Fraction operator+(const Fraction& f) const {

            return Fraction(numerator \* f.denominator + f.numerator \* denominator,denominator \* f.denominator);

        }

        Fraction operator-(const Fraction& f) const {

            return Fraction(numerator \* f.denominator - f.numerator \* denominator, denominator \* f.denominator);

        }

        Fraction operator\*(const Fraction& f) const {

            return Fraction(numerator \* f.numerator, denominator \* f.denominator);

        }

        Fraction operator/(const Fraction& f) const {

            return Fraction(numerator \* f.denominator, denominator \* f.numerator);

        }

        friend ostream& operator<<(ostream& out, const Fraction& f);

};

ostream& operator<<(ostream& out, const Fraction& f) {

    out << f.numerator << "/" << f.denominator;

    return out;

}

int main() {

    Fraction f1(1, 2), f2(1, 4);

    cout << "f1: 1/2, f2: 1/4" << endl;

    cout << "f1 + f2 = " << (f1 + f2) << endl;

    cout << "f1 - f2 = " << (f1 - f2) << endl;

    cout << "f1 \* f2 = " << (f1 \* f2) << endl;

    cout << "f1 / f2 = " << (f1 / f2) << endl;

    return 0;

}

A screenshot of a computer

AI-generated content may be incorrect.