**4 - Amaliy topshiriq**

*Iqbolshoh Ilhomjonov*

1. Shaxsiy a'zolar hisob raqami, balans va displayBalance() umumiy usuli bilan BankAccount nomli sinf yarating. BankAccount ob'ekti balansiga mablag' qo'shish imkonini beruvchi depozitAmount() nomli do'st funksiyasini loyihalash.

#include <iostream>

using namespace std;

class BankAccount

{

private:

    float balance;

public:

    BankAccount() : balance(0) {}

    void displayBalance()

    {

        cout << " Balance : " << balance << " $ " << endl;

    }

    friend BankAccount depozitAmount(BankAccount &Bank, float newbalance);

};

BankAccount depozitAmount(BankAccount &Bank, float newbalance)

{

    Bank.balance = newbalance;

}

int main()

{

    BankAccount obj1;

    obj1.displayBalance();

    depozitAmount(obj1, 777000);

    obj1.displayBalance();

    return 0;

}

// Natija



1. "uzunlik" va "kenglik" xususiy a'zolari bilan "To'rtburchak" nomli sinfni va maydonni hisoblash uchun "calculateArea()" umumiy usulini loyihalash. Uzunligi 5 va eni 10 boʻlgan toʻrtburchakning maydonini hisoblash uchun “Rectangle” sinfining anonim obyektini yarating

#include <iostream>

using namespace std;

class Tortburchak

{

private:

    int a;

    int b;

    int area;

public:

    Tortburchak() : a(0), b(0), area(0) {}

    void Print()

    {

        cout << " a = " << a << endl;

        cout << " b = " << b << endl;

        cout << " Area = " << area << endl;

    }

    friend Tortburchak calculateArea(Tortburchak &Shakl, int a, int b);

};

Tortburchak calculateArea(Tortburchak &Shakl, int a, int b)

{

    Shakl.a = a;

    Shakl.b = b;

    Shakl.area = a \* b;

}

int main()

{

    Tortburchak Rectangle;

    Rectangle.Print();

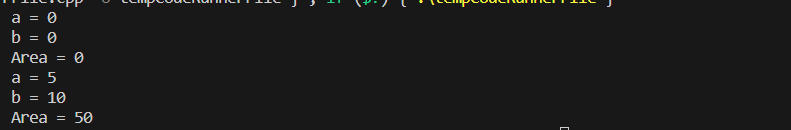
    calculateArea(Rectangle, 5, 10);

    Rectangle.Print();

    return 0;

}

// Natija



1. "haqiqiy" va "mavhum" shaxsiy a'zolarga ega bo'lgan murakkab sonlarni ifodalovchi "ComplexNumber" nomli sinfni aniqlang. Ikkita "ComplexNumber" ob'ektini parametr sifatida qabul qiladigan va ularning yig'indisini qaytaradigan "addComplex()" do'st funksiyasini yarating. Qo'shish amalini bajarish uchun anonim ob'ektdan foydalaning

#include <iostream>

using namespace std;

class ComplexNumber {

private:

    double real;

    double imaginary;

public:

    ComplexNumber(double realPart = 0, double imaginaryPart = 0)

        : real(realPart), imaginary(imaginaryPart) {}

    ComplexNumber addComplex(const ComplexNumber& other) const {

        return ComplexNumber(real + other.real, imaginary + other.imaginary);

    }

    void print() const {

        cout << "Real: " << real << ", Imaginary: " << imaginary << endl;

    }

};

int main() {

    ComplexNumber firstComplex(2.5, 3.7);

    ComplexNumber secondComplex(1.8, 4.2);

    ComplexNumber result = firstComplex.addComplex(secondComplex);

    cout << "Sum of Complex Numbers: ";

    result.print();

    return 0;

}

// Natija



1. `x` va `y` shaxsiy a`zolari bilan `Point` nomli sinf yarating. Ikkita “Point” obyekti orasidagi masofani hisoblaydigan “calculateDistance()” funksiyasini yarating. Masofa formulasi yordamida ushbu funktsiyani bajaring.

#include <iostream>

#include <cmath>

using namespace std;

class Point

{

private:

    double x;

    double y;

    double distance;

public:

    Point() : x(0), y(0), distance(0) {}

    void Print()

    {

        cout << " x = " << x << endl;

        cout << " y = " << y << endl;

        cout << " distance = " << distance << endl;

    }

    friend Point calculateDistance(Point &Nuqta, double x, double y);

};

Point calculateDistance(Point &Nuqta, double x, double y)

{

    Nuqta.x = x;

    Nuqta.y = y;

    Nuqta.distance = abs(x - y);

}

int main()

{

    Point obj1;

    obj1.Print();

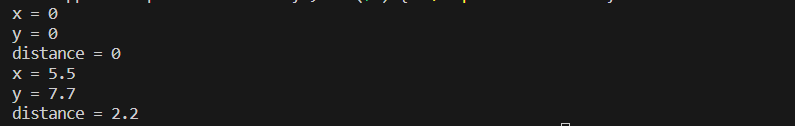
    calculateDistance(obj1, 5.5, 7.7);

    obj1.Print();

    return 0;

}

// Natija



1. “Talaba” va “Kutubxona” nomli ikkita sinf yarating, bu yerda “Talaba” talabalar ma’lumotlarini saqlaydi va “Kutubxona” kitoblar mavjudligini boshqaradi. “Kutubxona” sinfi talabaning qarz olish tarixiga kirishi va yangilanishi mumkin boʻlgan doʻstlar sinfi munosabatlarini yarating.

#include <iostream>

using namespace std;

class Kutubxona

{

private:

    string name;

    int qarz;

public:

    Kutubxona(string n, int a)

    {

        name = n;

        qarz = a;

    }

    void print()

    {

        cout << "Name: " << name << endl;

        cout << "Qarz: " << qarz << endl;

        cout << endl;

    }

    friend class Talaba;

};

class Talaba

{

private:

    int newData;

public:

    Talaba(int n)

    {

        newData = n;

    }

    void SetFunc(Kutubxona &human)

    {

        human.qarz = newData;

    }

    void print()

    {

        cout << "Yangi malumot ";

        cout << "Qarz: " << newData << endl;

    }

};

int main()

{

    Kutubxona obj1("SamDU kutubxonasi", 222);

    obj1.print();

    Talaba obj2(177);

    obj2.SetFunc(obj1);

    obj2.print();

    return 0;

}

// Natija



1. "Id" va "ish haqi" shaxsiy a'zolari bilan "Xodim" nomli sinfni amalga oshiring. Saralash algoritmidan foydalanib, "Xodim" ob'ektlari massivini ularning ish haqi asosida saralash uchun anonim ob'ekt yarating.

#include <iostream>

#include <algorithm>

using namespace std;

class Xodim

{

public:

    int ID;

    float Maosh;

    void Print() const

    {

        cout << " ID = " << ID << " Maosh = " << Maosh << endl;

    }

};

bool operator<(const Xodim &a, const Xodim &b)

{

    return a.Maosh < b.Maosh;

}

int main()

{

    const int size = 3;

    Xodim xodimlar[size];

    for (int i = 0; i < size; ++i)

    {

        cout << i + 1 << " - Xodim ID va maoshni kiriting: ";

        cin >> xodimlar[i].ID >> xodimlar[i].Maosh;

    }

    cout << "Saralashdan oldin:" << endl;

    for (const Xodim &xodim : xodimlar)

    {

        xodim.Print();

    }

    sort(xodimlar, xodimlar + size);

    cout << "Saralashdan so'ng:" << endl;

    for (const Xodim &xodim : xodimlar)

    {

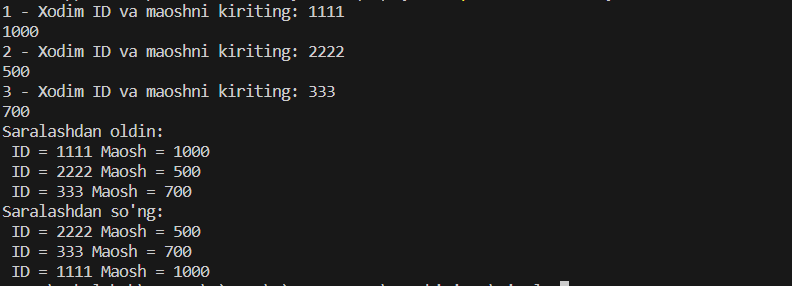
        xodim.Print();

    }

    return 0;

}

// Natija



1. 2D matritsani ifodalovchi “Matrix” nomli sinfni ishlab chiqing. Ikkita “Matrisa” obyekti oʻrtasida matritsalarni koʻpaytirishni amalga oshiradigan “multiplyMatrices()” nomli doʻst funksiyasini loyihalashtiring.

#include <iostream>

#include <vector>

using namespace std;

class Matrix

{

private:

    vector<vector<int>> data;

    int rows, cols;

public:

    Matrix(int r, int c) : rows(r), cols(c)

    {

        data.resize(rows, vector<int>(cols, 0));

    }

    void fillMatrix()

    {

        cout << "Matritsa elementlarini qator bo'yicha kiriting:\n";

        for (int i = 0; i < rows; ++i)

        {

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

            {

                cin >> data[i][j];

            }

        }

    }

    void displayMatrix() const

    {

        for (int i = 0; i < rows; ++i)

        {

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

            {

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

            }

            cout << "\n";

        }

    }

    friend Matrix multiplyMatrices(const Matrix &m1, const Matrix &m2);

};

Matrix multiplyMatrices(const Matrix &m1, const Matrix &m2)

{

    int rows1 = m1.rows;

    int cols1 = m1.cols;

    int cols2 = m2.cols;

    Matrix result(rows1, cols2);

    for (int i = 0; i < rows1; ++i)

    {

        for (int j = 0; j < cols2; ++j)

        {

            int sum = 0;

            for (int k = 0; k < cols1; ++k)

            {

                sum += m1.data[i][k] \* m2.data[k][j];

            }

            result.data[i][j] = sum;

        }

    }

    return result;

}

int main()

{

    int rows, cols;

    cout << "Matritsa o'lchamini kiriting (qator ustunlar): ";

    cin >> rows >> cols;

    Matrix matrix1(rows, cols);

    Matrix matrix2(rows, cols);

    cout << "Matritsa 1 elementlarini kiriting:\n";

    matrix1.fillMatrix();

    cout << "Matritsa 2 elementlarini kiriting:\n";

    matrix2.fillMatrix();

    Matrix result = multiplyMatrices(matrix1, matrix2);

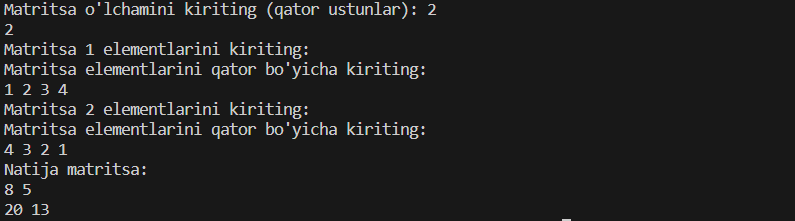
    cout << "Natija matritsa:\n";

    result.displayMatrix();

    return 0;

}

// Natija



1. "Foydalanuvchi nomi" shaxsiy a'zosi bilan "User" nomli sinf yarating. Foydalanuvchi autentifikatsiyasini boshqarish uchun “Foydalanuvchi”ga do‘st sinf sifatida “Autentifikatsiya” nomli sinfni ishlab chiqing. Kiritilgan foydalanuvchi nomi saqlangan foydalanuvchi nomiga mos kelishini tekshirish orqali autentifikatsiya jarayonini simulyatsiya qiling.

#include <iostream>

using namespace std;

class User

{

private:

    string name;

public:

    User() : name("Iqbolshoh") {}

    void Print()

    {

        cout << " name : " << name << endl;

    }

    friend User Autentifikatsiya(User &user, string newname);

};

User Autentifikatsiya(User &user, string newname)

{

    if (user.name == newname)

    {

        cout << " Ma'lomot bir xil " << endl;

    }

    else

    {

        cout << " Ma'lumot Bir xil emas " << endl;

    }

}

int main()

{

    User obj1;

    obj1.Print();

    Autentifikatsiya(obj1, "Iqbolshoh");

    return 0;

}

// Natija



1. Jurnallarni saqlash uchun shaxsiy a'zolar bilan "Logger" sinfini loyihalash. Muayyan hodisa uchun jurnalga yozuv yozish va keyin jurnallarni ko'rsatish uchun "Logger" sinfining anonim ob'ektini yarating.

#include <iostream>

#include <functional>

using namespace std;

class Logger

{

private:

    string malumot;

public:

    Logger() : malumot() {}

    void SetMalumot(const function<string()> &inputFunction)

    {

        malumot = inputFunction();

    }

    void Print()

    {

        cout << " malumot : " << malumot << endl;

    }

};

int main()

{

    Logger obj1;

    obj1.SetMalumot([]() {

        cout << "Malumotni kiriting: ";

        string input;

        cin >> input;

        return input;

    });

    obj1.Print();

    return 0;

}

// Natija



1. `x` va `y` shaxsiy a`zolari bilan `Point` nomli sinf yarating. Ikkita “Point” obyekti orasidagi masofani hisoblaydigan “calculateDistance()” nomli do‘st funksiyasini yarating. Bu funksiyani Pifagor teoremasidan foydalanib bajaring.

#include <iostream>

#include <cmath>

using namespace std;

class Point

{

private:

    double x;

    double y;

    double distance;

public:

    Point() : x(0), y(0), distance(0) {}

    void Print()

    {

        cout << " x = " << x << endl;

        cout << " y = " << y << endl;

        cout << " distance = " << distance << endl;

    }

    friend Point calculateDistance(Point &Nuqta, double x, double y);

};

Point calculateDistance(Point &Nuqta, double x, double y)

{

    Nuqta.x = x;

    Nuqta.y = y;

    Nuqta.distance = sqrt(x \* x + y \* y);

}

int main()

{

    Point obj1;

    obj1.Print();

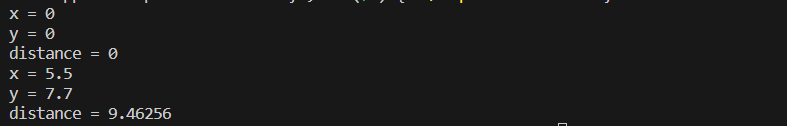
    calculateDistance(obj1, 5.5, 7.7);

    obj1.Print();

    return 0;

}

// Natija



1. "O'qituvchi" va "Bo'lim" ikkita sinf yarating, bu yerda "O'qituvchi" o'qituvchi ma'lumotlarini saqlaydi va "Bo'lim" bo'lim tafsilotlarini boshqaradi. "Bo'lim" sinfi o'qituvchi ma'lumotlariga kirishi va ko'rsatishi mumkin bo'lgan do'stlar sinfi munosabatlarini yarating.

#include <iostream>

using namespace std;

class Oqituvchi

{

private:

    string name;

    string surname;

public:

    Oqituvchi(string n, string a)

    {

        name = n;

        surname = a;

    }

    void print()

    {

        cout << "Name: " << name << endl;

        cout << "Surname: " << surname << endl;

        cout << endl;

    }

    friend class Bolim;

};

class Bolim

{

private:

    string newname;

public:

    Bolim(string n)

    {

        newname = n;

    }

    void SetFunc(Oqituvchi &human)

    {

        human.name = newname;

    }

    void print()

    {

        cout << "Yangi malumot ";

        cout << "Name: " << newname << endl;

    }

};

int main()

{

    Oqituvchi obj1("Iqbolshoh", "Ilhomjonov");

    obj1.print();

    Bolim obj2("Iqbolshoh\_777");

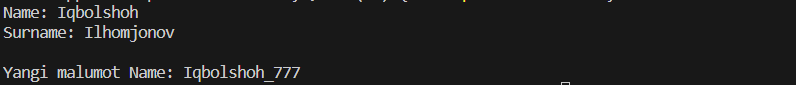
    obj2.SetFunc(obj1);

    obj2.print();

    return 0;

}

// Natija



1. Selsiy bo'yicha haroratni saqlash uchun shaxsiy a'zolar bilan "Temperatura" nomli sinfni amalga oshiring. Haroratni Selsiydan Farengeytga aylantirish uchun “Temperatura” sinfining anonim obyektini yarating.

#include <iostream>

using namespace std;

class Temperatura

{

private:

    float selsiy;

    float faradey;

public:

    Temperatura() : faradey(0) {}

    void Set(float b)

    {

        selsiy = b;

    }

    void Print()

    {

        cout << " Faradey = " << faradey << endl;

    }

    friend Temperatura calculatefaradey(Temperatura &harorat);

};

Temperatura calculatefaradey(Temperatura &harorat)

{

    harorat.faradey = (harorat.selsiy \* 9 / 5) + 32;

    return harorat;

}

int main()

{

    Temperatura obj1;

    obj1.Set(30);

    obj1.Print();

    obj1 = calculatefaradey(obj1);

    obj1.Print();

    return 0;

}

// Natija



1. Mahsulot tafsilotlari uchun shaxsiy a'zolar bilan "Shop" sinfini loyihalash. Do‘kondan mahsulotlarga kirishi va xarid qilishi mumkin bo‘lgan “Mijoz” nomli do‘stlar sinfini yarating. Ushbu sinflar yordamida xarid qilish stsenariysini simulyatsiya qiling.

#include <iostream>

using namespace std;

class Shop

{

private:

    string name;

    string xaridor;

public:

    Shop(string n, string a)

    {

        name = n;

        xaridor = a;

    }

    void print()

    {

        cout << "Name: " << name << endl;

        cout << "xaridor: " << xaridor << endl;

        cout << endl;

    }

    friend class Mijoz;

};

class Mijoz

{

private:

    string newname;

public:

    Mijoz(string n)

    {

        newname = n;

    }

    void SetFunc(Shop &human)

    {

        human.xaridor = newname;

    }

    void print()

    {

        cout << "Yangi malumot ";

        cout << "Name: " << newname << endl;

    }

};

int main()

{

    Shop obj1("Market", "Iqbolshoh");

    obj1.print();

    Mijoz obj2("Azizjon");

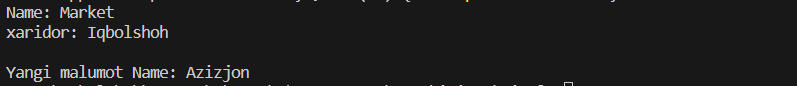
    obj2.SetFunc(obj1);

    obj2.print();

    return 0;

}

// Natija



1. Butun qiymatlar massivini saqlay oladigan `DataList` nomli sinfni loyihalash. “DataList” obyektidagi qiymatlarning o‘rtacha qiymatini hisoblaydigan “calculateAverage()” nomli do‘st funksiyasini yarating.

#include <iostream>

using namespace std;

class DataList

{

private:

    int \*data;

    int size;

public:

    DataList(int s) : size(s)

    {

        data = new int[size];

        for (int i = 0; i < size; ++i)

        {

            data[i] = 0;

        }

    }

    ~DataList()

    {

        delete[] data;

    }

    void addData(int value, int index)

    {

        if (index >= 0 && index < size)

        {

            data[index] = value;

        }

        else

        {

            cout << "Invalid index!" << endl;

        }

    }

    double calculateAverage()

    {

        if (size == 0)

        {

            cout << "DataList is empty." << endl;

            return 0.0;

        }

        int sum = 0;

        for (int i = 0; i < size; ++i)

        {

            sum += data[i];

        }

        return static\_cast<double>(sum) / size;

    }

};

int main()

{

    DataList dataList(5);

    dataList.addData(10, 0);

    dataList.addData(20, 1);

    dataList.addData(30, 2);

    dataList.addData(40, 3);

    dataList.addData(50, 4);

    cout << "Average: " << dataList.calculateAverage() << endl;

    return 0;

}

// Natija



1. "Shaxs" va "Manzil" sinflarini yarating. Ular o'rtasida do'stlar sinfi munosabatlarini yarating, bu erda "Manzil" klassi "Shaxs"ning aloqa ma'lumotlariga kirishi va ko'rsatishi mumkin.

#include <iostream>

using namespace std;

class Shaxs

{

private:

    string name;

    string manzil;

public:

    Shaxs(string n)

    {

        name = n;

    }

    void print()

    {

        cout << "Name: " << name << endl;

    }

    friend class Manzil;

};

class Manzil

{

private:

    string newname;

public:

    Manzil(string n)

    {

        newname = n;

    }

    void SetFunc(Shaxs &human)

    {

        human.manzil = newname;

    }

    void print()

    {

        cout << "Manzil: " << newname << endl;

    }

};

int main()

{

    Shaxs obj1("Iqbolshoh");

    obj1.print();

    Manzil obj2("Samarqand");

    obj2.SetFunc(obj1);

    obj2.print();

    return 0;

}

// Natija



1. “Valyuta kurslarini saqlash uchun shaxsiy a'zolar bilan "CurrencyConverter" sinfini yarating. Mablag'ni bir valyutadan boshqasiga o'tkazish uchun "CurrencyConverter" sinfining anonim ob'ektini yarating.

#include <iostream>

using namespace std;

class CurrencyConverter

{

private:

    int balance;

public:

    CurrencyConverter(int a) : balance(a) {}

    void displayBalance()

    {

        cout << " Balance : " << balance << " $ " << endl;

    }

    void displayBalanc()

    {

        cout << " Balance : " << balance << " sum " << endl;

    }

    friend void depozitAmount(CurrencyConverter &Bank, int newbalance);

};

void depozitAmount(CurrencyConverter &Bank, int newbalance)

{

    Bank.balance = newbalance \* 12918;

}

int main()

{

    int dollar = 500;

    CurrencyConverter obj1(dollar);

    obj1.displayBalance();

    depozitAmount(obj1, dollar);

    obj1.displayBalanc();

    return 0;

}

// Natija



1. Hisob tafsilotlari uchun shaxsiy a'zolar bilan "Bank" sinfini yarating. Hisob ma’lumotlariga kirishi va tranzaktsiyalarni amalga oshirishi mumkin bo‘lgan “Mijoz” nomli do‘stlar sinfini yarating. Ushbu sinflar yordamida bank stsenariysini simulyatsiya qiling.

#include <iostream>

using namespace std;

class Bank

{

private:

    float balance;

public:

    Bank() : balance(0) {}

    void displayBalance()

    {

        cout << " Balance : " << balance << " $ " << endl;

    }

    friend Bank depozitAmount(Bank &Bank, float newbalance);

};

Bank depozitAmount(Bank &Bank, float newbalance)

{

    Bank.balance = newbalance;

}

int main()

{

    Bank obj1;

    obj1.displayBalance();

    depozitAmount(obj1, 777000);

    obj1.displayBalance();

    return 0;

}

// Natija



1. Selsiy bo'yicha haroratni saqlash uchun shaxsiy a'zolar bilan "Harorat" sinfini amalga oshiring. Harorat qiymatini Selsiydan Kelvinga aylantirish uchun "Temperatura" sinfining anonim ob'ektidan foydalaning

#include <iostream>

using namespace std;

class Haraorat

{

private:

    float selsiy;

    float kelvin;

public:

    Haraorat() : kelvin(0) {}

    void Set(float b)

    {

        selsiy = b;

    }

    void Print()

    {

        cout << " Kelvin = " << kelvin << endl;

    }

    friend Haraorat calculatekelvin(Haraorat &harorat, float selsiy);

};

Haraorat calculatekelvin(Haraorat &harorat, float selsiy)

{

    harorat.kelvin = selsiy + 273.15;

}

int main()

{

    Haraorat obj1;

    obj1.Set(30);

    obj1.Print();

    calculatekelvin(obj1, 30);

    obj1.Print();

    return 0;

}

// Natija

