Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

Кафедра кібербезпеки та програмного забезпечення

Дисципліна: Об’єктно-орієнтоване програмування

**Лабораторна робота №2**

**Тема:** **«**Конструктори і деструктори**»**

|  |
| --- |
| Виконав: ст. гр. КБ-23 |
| Захарченко А.О. |
| Перевірив: викладач  Козірова Наталія Леонідівна |
|  |

Кропивницький

2024

**Тема:** Конструктори і деструктори

**Мета:** ознайомитись з основними поняттями конструктор і деструктор в ООП та навчитись їх програмно реалізовувати мовою С++.

**Завдання:**

**Завдання 1**

1. Реалізуйте конструктор за замовчуванням, конструктор з параметрами та копіюючий конструктор для вашого класу з лабораторної роботи 1.

* Конструктор за замовчуванням має встановлювати значення полів за замовчуванням.
* Конструктор з параметрами має приймати значення для кожного поля.
* Копіюючий конструктор має копіювати значення полів з іншого об'єкта класу.

2. Реалізуйте деструктор для класу. Деструктор має виводити повідомлення про знищення об'єкта.

3. У функції main створіть об'єкт за допомогою конструктора за замовчуванням та виведіть значення його полів.

4. Створіть новий об'єкт за допомогою конструктора з параметрами та встановіть значення для полів. Виведіть значення полів цього об'єкта.

5. Створіть ще один об'єкт і скопіюйте значення полів з першого об'єкта за допомогою копіюючого конструктора. Виведіть значення полів цього об'єкта.

6. Завершіть функцію main, що призведе до виходу з області видимості створених об'єктів і виклику їх деструкторів. Переконайтесь, що повідомлення про знищення об'єктів виводяться

**Завдання 2:**

**Варіант 3**

Розробіть клас «Квадратна матриця» – Matrix. Клас повинен містити конструктори. Реалізуйте методи для додавання, віднімання, множення матриць та обчислення норми матриці. Створіть масив об'єктів класу Matrix і передайте його у функцію, яка змінює 𝑖-ту матрицю шляхом піднесення її до квадрату. У головній програмі виведіть результат.

**Реалізація завдання 1:**

**Файл Person.cpp:**

#include "Person.h"

using namespace std;

void Person::setname(const string name) {

this->name = name;

}

void Person::setage(int age) {

this->age = age;

}

void Person::setaddress(const string address) {

this->address = address;

}

string Person::getname() {

return name;

}

int Person::getage() {

return age;

}

string Person::getaddress() {

return address;

}

Person::Person() : name("no\_data"), age(0), address("no\_data") { //Конструктор за замовчуванням

cout << "Usual constr " << endl; //Теж

}

Person::Person(const string name, int age, string address) : name(name), age(age), address(address) {

cout << "Paramentr constr" << endl;

}

Person::Person(const Person& other) : name(other.name), age(other.age), address(other.address) {

cout << "Copy constr" << endl;

}

;

Person::~Person() {

cout << "Destructor for this one: " << endl;

}

**Файл Person.h:**

#pragma once

#ifndef PERSON\_H

#define PERSON\_H

#include <string>

#include <iostream>

using namespace std;

class Person

{

private:

string name;

int age;

string address;

public:

Person();

Person(string name, int age, string address);

Person(const Person& other);

string getname();

int getage();

string getaddress();

void setname(string name);

void setage(int age);

void setaddress(string address);

~Person();

};

#endif

**Файл LR-2.cpp:**

#include <iostream>

#include "Person.h"

using namespace std;

int main() {

Person person;

string name;

int age;

string address;

cout << "Name: " << endl;

cin >> name;

person.setname(name);

cout << "Age: " << endl;

cin >> age;

person.setage(age);

cout << "Address: " << endl;

cin >> address;

person.setaddress("address");

cout << "Name: " << person.getname() << endl;

cout << "Age: " << person.getage() << endl;

cout << "Address: " << person.getaddress() << endl;

Person person1("Josh", 35, "Luvabur");

{

cout << "Name: " << person1.getname() << endl;

cout << "Age: " << person1.getage() << endl;

cout << "Address: " << person1.getaddress() << endl;

}

Person person3 = person1;

cout << "Name: " << person3.getname() << endl;

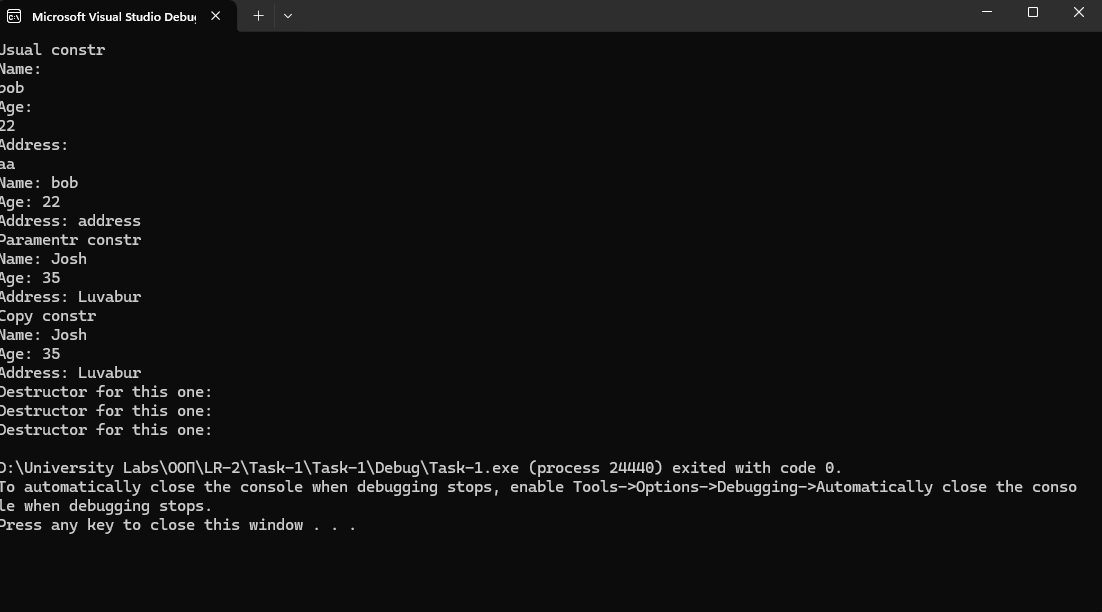
cout << "Age: " << person3.getage() << endl;

cout << "Address: " << person3.getaddress() << endl;

return 0;

}

**Результат**

****

**Реалізація завдання 2:**

**Файл Matrix.cpp:**

#include "Matrix.h"

#include <iostream>

#include <stdexcept>

using namespace std;

// Конструктор за замовчуванням

Matrix::Matrix(int size) : size(size), elements(size, vector<double>(size, 0.0)) {}

// Конструктор з параметрами

Matrix::Matrix(int size, const vector<vector<double>>& values)

: size(size), elements(values) {

if (values.size() != size || values[0].size() != size) {

throw invalid\_argument("Matrix dimensions do not match the specified size.");

}

}

// Метод для додавання матриць

Matrix Matrix::add(const Matrix& other) const {

if (size != other.size) {

throw invalid\_argument("Matrix sizes do not match.");

}

Matrix result(size);

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

for (int j = 0; j < size; ++j) {

result.elements[i][j] = elements[i][j] + other.elements[i][j];

}

}

return result;

}

// Метод для віднімання матриць

Matrix Matrix::subtract(const Matrix& other) const {

if (size != other.size) {

throw invalid\_argument("Matrix sizes do not match.");

}

Matrix result(size);

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

for (int j = 0; j < size; ++j) {

result.elements[i][j] = elements[i][j] - other.elements[i][j];

}

}

return result;

}

// Метод для множення матриць

Matrix Matrix::multiply(const Matrix& other) const {

if (size != other.size) {

throw invalid\_argument("Matrix sizes do not match.");

}

Matrix result(size);

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

for (int j = 0; j < size; ++j) {

result.elements[i][j] = 0;

for (int k = 0; k < size; ++k) {

result.elements[i][j] += elements[i][k] \* other.elements[k][j];

}

}

}

return result;

}

// Метод для обчислення норми матриці

double Matrix::norm() const {

double sum = 0.0;

for (const auto& row : elements) {

for (double value : row) {

sum += value \* value;

}

}

return sqrt(sum);

}

// Метод для піднесення матриці до квадрату

Matrix Matrix::square() const {

return multiply(\*this);

}

// Метод для виводу матриці на екран

void Matrix::print() const {

for (const auto& row : elements) {

for (double value : row) {

cout << value << " ";

}

cout << endl;

}

}

// Доступ до елементів матриці

double& Matrix::at(int i, int j) {

if (i < 0 || i >= size || j < 0 || j >= size) {

throw out\_of\_range("Index out of range.");

}

return elements[i][j];

}

double Matrix::at(int i, int j) const {

if (i < 0 || i >= size || j < 0 || j >= size) {

throw out\_of\_range("Index out of range.");

}

return elements[i][j];

}

**Файл Matrix.h:**

#ifndef MATRIX\_H

#define MATRIX\_H

#include <iostream>

#include <vector>

#include <cmath>

class Matrix {

private:

int size;

std::vector<std::vector<double>> elements;

public:

// Конструктор за замовчуванням

Matrix(int size = 3);

// Конструктор з параметрами

Matrix(int size, const std::vector<std::vector<double>>& values);

// Додати матрицю

Matrix add(const Matrix& other) const;

// Відняти матрицю

Matrix subtract(const Matrix& other) const;

// Помножити матрицю

Matrix multiply(const Matrix& other) const;

// Обчислити норму матриці

double norm() const;

// Піднести матрицю до квадрату

Matrix square() const;

// Вивести матрицю на екран

void print() const;

// Доступ до елементів матриці

double& at(int i, int j);

double at(int i, int j) const;

};

#endif // MATRIX\_H

**Файл LR-2.cpp:**

#include <iostream>

#include "Matrix.h"

using namespace std;

void squareMatrix(Matrix& mat) {

mat = mat.square();

}

int main() {

// Створення масиву матриць

const int numMatrices = 2;

Matrix matrices[numMatrices] = {

Matrix(2, {{1, 2}, {3, 4}}),

Matrix(2, {{5, 6}, {7, 8}})

};

// Виведення початкових матриць

cout << "Initial matrices:" << endl;

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

cout << "Matrix " << i + 1 << ":" << endl;

matrices[i].print();

}

// Піднесення кожної матриці до квадрата

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

squareMatrix(matrices[i]);

}

// Виведення результатів

cout << "Matrices squared:" << endl;

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

cout << "Matrix " << i + 1 << ":" << endl;

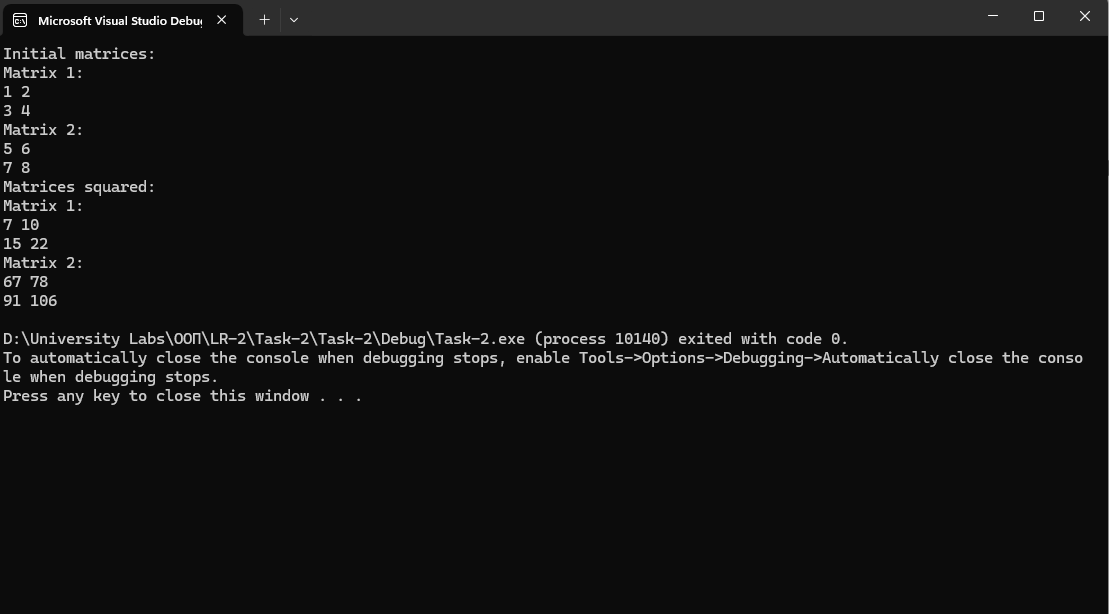
matrices[i].print();

}

return 0;

}

**Результат:**

****

**Висновок:** отже виконавши ЛР2, було покращено навичики написання коду на мові програмування С++, повторено основну структуру коду, вивченно та опрацьовано поняття конструкторів типу:

* Конструктор за замовчуванням
* Конструктор з параметрами
* Копіювальний конструктор
* Деструктор.