Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи №3 з дисципліни

«Основи програмування-2.

Модульне програмування»

«Класи та об'єкти»

Варіант 19

Виконав студент: ІП-11 Лисенко Андрій Юрійович

Перевірила: Вітковська Ірина Іванівна

Київ 2021

**Лабораторна робота №3**

**Класи та об'єкти**

**Мета роботи**

Вивчити механізми створення і використання класів та об'єктів

**Індивідуальне завдання**

**Варіант 19**

Розробити клас "квадратна матриця" вказаної розмірності. Створити масив об'єктів даного класу. Визначити матрицю з найбільшою сумою діагональних елементів.

**Постановка задачі**

Створимо клас SquareMatrix для роботи з квадратними матриць. Об'єкти цього класу матимуть єдиний атрибут size, що вказуватиме на розмірність матриці. Також цей клас матиме конструктор та декілька методів, а саме show() та sum\_diagonal(). Ці два методи будуть виводити матрицю на екран та знаходити суму діагональних елементів відповідно. Також створимо клас для масиву матриць: SquareMatrixArray. Атрибут matrices буде мати тим std::vector<SquareMatrix>, тобто клас SquareMatrix буде мати відношеня агрегації до класу SquareMatrixArray. Розробимо метод find\_greatest\_diagonal\_sum(), що знайде матрицю з максимальної суму діагональних елементів.

**Побудова математичної моделі**

Складемо таблицю змінних

|  |  |  |  |
| --- | --- | --- | --- |
| Змінна | Тип | Ім’я | Призначення |
| Розмірність | *int* | *size* | Зберігати розмірність матриць |
| Довжина | *int* | *len* | Зберігати довжину масиву матриць |
| Масив матриць | *SquareMatrixArray* | *matrix\_arr* | зберігати покажчик на масив матриць |
| Найбільша | *SquareMatrix* | *greatest* | Зберігати об'єкт матриці з найбільшою сумую діагональних елементів |

**Псевдокод**

**Основна програма**

input size, len

matrix\_arr = SquareMatrixArray(len, size)

output matrix\_arr

greatest = matrix\_arr.find\_greatest\_diagonal\_sum()

output greatest

**Конструктор *SquareMatrix***

for row from 0 to size:

for col from 0 to size

vec.push\_back(random(-100, 100))

elements.push\_back(vec)

**Метод *SquareMatrix: show***

for row in matrices:

for element in row:

show element

**Метод *SquareMatrix: sum\_diagonal***

sum = 0

for i from 0 to size:

sum += elements[i][i]

return sum

**Конструктор *SquareMatrixArray***

g\_pos = 0

g\_val = matrices[0].sum\_diagonal()

for i from i to len

current = matrices[i].sum\_diagonal()

if current > g\_val

g\_pos = i

g\_val = current

return g\_pos

**Метод *SquareMatrixArray: show***

i = 0

for matrix in matrices:

show matrix

**Метод *SquareMatrixArray: get\_element***

return matrices[pos]

**Метод *SquareMatrixArray: find\_greatest\_diagonal\_sum***

g\_pos = 0

g\_val = matrices[0].sum\_diagonal()

for i from i to len

current = matrices[i].sum\_diagonal()

if current > g\_val

g\_pos = i

g\_val = current

return g\_pos

**Код на С++**

**main.cpp**

#include <iostream>

#include "matrix.h"

int main()

{

int size;

std::cout << "Enter matrix size: "; std::cin >> size;

int len;

std::cout << "Enter length of matrix array: "; std::cin >> len;

SquareMatrixArray matrix\_arr(len, size);

std::cout << "Generated matrices:\n";

matrix\_arr.show();

int greatest = matrix\_arr.find\_greatest\_diagonal\_sum();

std::cout << "The " << greatest + 1 << " matrix has the greatest diagonal sum:\n\n";

matrix\_arr.get\_element(greatest).show();

std::cout << "Sum: " << matrix\_arr.get\_element(greatest).sum\_diagonal() << '\n';

system("pause");

return 0;

}

**matrix.cpp**

#include "matrix.h"

#include <windows.h>

#include <string>

SquareMatrix::SquareMatrix(int size) : size(size)

{

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

std::vector<int> vec;

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

vec.push\_back(rand() % 201 - 100);

elements.push\_back(vec);

}

}

void SquareMatrix::show()

{

for (std::vector<int> row : elements) {

for (int element : row)

std::cout << std::setw(5) << element;

std::cout << '\n';

}

std::cout << '\n';

}

int SquareMatrix::sum\_diagonal()

{

int sum = 0;

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

sum += elements[i][i];

return sum;

}

SquareMatrixArray::SquareMatrixArray(int len, int size) : len(len), size(size)

{

srand((unsigned int)time(NULL));

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

matrices.push\_back(SquareMatrix(size));

}

void SquareMatrixArray::show()

{

int i = 1;

for (SquareMatrix matrix : matrices) {

std::cout << i++ << ")\n";

matrix.show();

}

std::cout << "\n";

}

SquareMatrix SquareMatrixArray::get\_element(int pos)

{

return matrices[pos];

}

int SquareMatrixArray::find\_greatest\_diagonal\_sum()

{

int g\_pos = 0;

int g\_val = matrices[0].sum\_diagonal();

int current;

for (int i = 1; i < len; i++)

{

current = matrices[i].sum\_diagonal();

if (current > g\_val) {

g\_pos = i;

g\_val = current;

}

}

return g\_pos;

}

**header.cpp**

#include <cstdlib>

#include <ctime>

#include <iostream>

#include <iomanip>

#include <vector>

class SquareMatrix

{

private:

int size;

std::vector<std::vector<int>> elements;

public:

SquareMatrix(int size);

void show();

int sum\_diagonal();

};

class SquareMatrixArray

{

private:

int len, size;

std::vector<SquareMatrix> matrices;

public:

SquareMatrixArray(int len, int size);

void show();

SquareMatrix get\_element(int pos);

int find\_greatest\_diagonal\_sum();

};

**Висновок**

Підчас виконання цієї роботи, я розробив практичні навички роботи з класами та об'єктами, а саме створення атрибутів та методів класів, виведення оголошення класу у окремий заголовчний файл та реалізацію в окремому cpp файлі. Також я знавчився створювати об'єкти класу та використовувати його методи*.* Отже, я вивчив механізми створення і використання класів та об'єктів.