Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний

інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки інформації

і управління

Звіт

з лабораторної роботи № 8 з дисципліни

«Основи програмування-1»

«Багатовимірні масиви»

Варіант 3

Виконав студент ІП-02, Бурков Михайло В’чеславович

(шифр, прізвище, ім'я, по батькові)

Перевірила \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

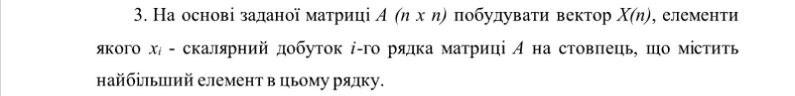
(прізвище, ім'я, по батькові)

Київ 2020

**Лабораторна робота №8**

**Мета роботи** – опанувати технологію використання двовимірних масивів даних (матриць), навчитися обробляти алгоритми та програми застосування матриць.

**Задача**:



**Блок-схеми**

Підпрограма initvec Підпрограма fill\_matrix\_random

void fill\_matrix\_random(int\* m, int n, int a, int b)

i = 0, n, 1

j = 0, n, 1

m[i \* n + j] = a + rand() % (a - b);

return

void initvec(int\* v, int num)

i = 0, num,1

v[i] =0

return

Підпрограма print\_vector

void print\_vector(int\* v, int n)

i = 0, n,1

return

v[i]

Початок

const int n = 5; int matrix[n][n]

fill\_matrix\_random(\*matrix, n, 1, 10)

print\_matrix(\*matrix, n)

int vec[n]

initvec(vec, n)

i = 0, n, 1

max = 0, idx = 0

find\_max\_element\_in\_row(matrix[i], n, max, idx)

j = 0, n, 1

vec[i] += matrix[i][j] \* matrix[j][idx]

print\_vector(vec, n)

Кінець

Основна програма

Підпрограма find\_max\_element\_in\_row

void find\_max\_element\_in\_row(int\* row, int n, int& val, int& idx)

val = row[0]

idx = 0

i = 0, n, 1

row[i] > val

val = row[i];

idx = i;

return

**Код на C++**

#include <iostream>

void fill\_matrix\_random(int\* , int , int , int );

void find\_max\_element\_in\_row(int\* , int , int& , int& );

void print\_matrix(int\* , int );

void print\_vector(int\* , int );

void initvec(int\* , int );

int main() {

setlocale(LC\_ALL, "Russian");

const int n = 5;

int matrix[n][n];

fill\_matrix\_random(\*matrix, n, 1, 10);

print\_matrix(\*matrix, n);

int vec[n];

initvec(vec, n);

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

int max{ 0 }, idx{ 0 };

find\_max\_element\_in\_row(matrix[i], n, max, idx);

std::cout << "Максимальний елемент " << i << "-го рядка = " << max << " с iндексом " << idx << std::endl;

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

vec[i] += matrix[i][j] \* matrix[j][idx];

}

std::cout << "Скалярний добуток " << i << "-го рядка та " << idx << "-го стовпця = " << vec[i] << std::endl;

}

print\_vector(vec, n);

}

void fill\_matrix\_random(int\* m, int n, int a, int b) {

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

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

m[i \* n + j] = a + rand() % (a - b);

}

}

}

void find\_max\_element\_in\_row(int\* row, int n, int& val, int& idx) {

val = row[0];

idx = 0;

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

if (row[i] > val) {

val = row[i];

idx = i;

}

}

}

void print\_matrix(int\* m, int n) {

std::cout << "Матриця:\n";

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

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

std::cout << m[i \* n + j] << " ";

}

std::cout << "\n";

}

}

void print\_vector(int\* v, int n) {

std::cout << "Вектор:\n";

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

std::cout << v[i] << " ";

}

std:: cout << "\n";

}

void initvec(int\* v, int num)

{

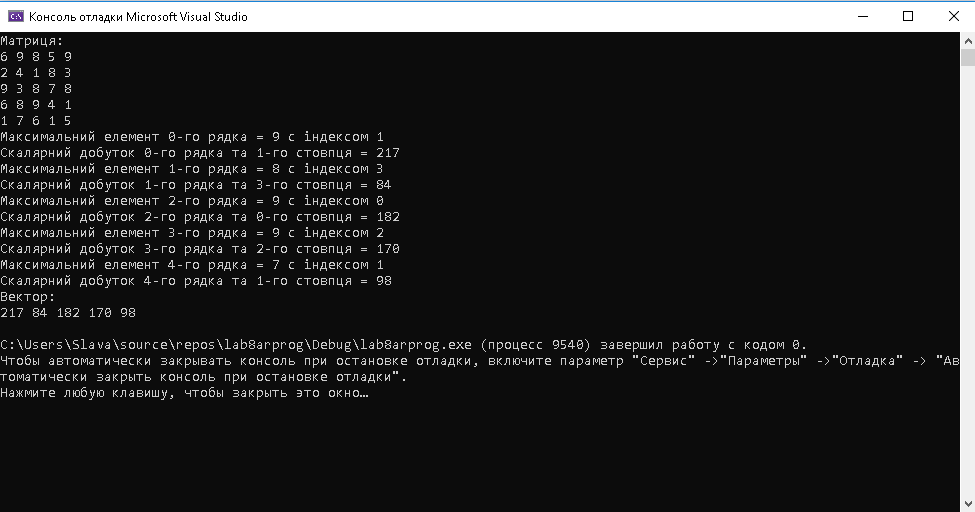
for (int i = 0; i < num; i++) {

v[i] = 0;

}

}

**Відеокопія результату на C++**



**Висновок**. Під час виконання цієї лабораторної роботи ми навчилися опрацьовувати алгоритми дії з багатовимірними масивами ( в нашому випадку матрицями ) та реалізовувати їх за допомогою програмної специфікації.