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 № 8 з дисципліни «Алгоритми та структури даних-1. Основи алгоритмізації»

«Дослідження алгоритмів перетворення матриць та отримання агрегатних значень»

Варіант 28

Виконав студент ІП-12, Сімчук Андрій Володимирович

(шифр, прізвище, ім'я, по батькові)

Перевірив \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

( прізвище, ім'я, по батькові)

Київ 2021

**Лабораторна робота 8**

**Дослідження алгоритмів перетворення матриць та отримання агрегатних значень**

**Мета -** дослідити підходи до пошуку та перетворення на матрицях та набути практичних навичок використання укладених керувальних дій повторення і їх з’єднання під час складання програмних специфікацій.

**I. Задача.**

Розробити алгоритм та написати програму, яка складається з наступних дій:

1. Опису та ініціювання змінної індексованого типу (двовимірний масив) 5 х 5 (цілий тип даних).

2. Створення нової змінної індексованого типу (одновимірний масив) та її

ініціювання значеннями (із додатних значень елементів головної діагоналі двовимірного масиву).

3. Відсортувати одновимірний масив обміном (бульбашкою) за спаданням.

**II. Розв’язання.**

Програмні специфікації запишемо у псевдокоді та графічній формі у вигляді блок-схеми.

*Крок 1.* Визначимо основні дії.

*Крок 2.* Деталізація ініціалізації змінних.

*Крок 3.* Деталізуємо дію заповнення двовимірного масиву.

*Крок 4.* Деталізуємо дію ініціалізації та заповнення одновимірного масиву.

*Крок 5.* Деталізуємо дію сортування одновимірного масиву бульбашкою за спаданням.

*Крок 6.* Деталізуємо дію виведення відсортованого одновимірного масиву.

***Побудова математичної моделі***

|  |  |  |  |
| --- | --- | --- | --- |
| **Змінна** | **Тип** | **Ім’я** | **Призначення** |
| Двовимірний масив розміром  5 х 5 | Індексований | A | Початкове значення |
| Одновимірний масив | Індексований | B | Проміжне значення, результат |
| Кількість елементів одновимірного масиву | Натуральний | k | Проміжне значення |
| Флажок | Логічний | flag | Проміжне значення |
| Показник лічильнику | Цілий | i | Проміжне значення |
| Показник лічильнику | Цілий | j | Проміжне значення |
| Додаткова змінна | Цілий | el | Проміжне значення |

***Псевдокод (основна програма):***

*Крок 1.*

**Початок**

ініціалізація змінних

заповнення двовимірного масиву

заповнення одновимірного масиву

сортування одновимірного масиву бульбашкою за спаданням

виведення відсортованого одновимірного масиву

**Кінець**

*Крок 2.*

**Початок**

k = 0

flag = true

A[5][5]

B[5]

заповнення двовимірного масиву

заповнення одновимірного масиву

сортування одновимірного масиву бульбашкою за спаданням

виведення відсортованого одновимірного масиву

**Кінець**

*Крок 3.*

**Початок**

k = 0

flag = true

A[5][5]

B[5]

**повторити**

**для** і **від** 0 **до** 5 **із кроком** 1

**повторити**

**для** j **від** 0 **до** 5 **із кроком** 1

**Введення A[i][j]**

**все повторити**

**все повторити**

заповнення одновимірного масиву

сортування одновимірного масиву бульбашкою за спаданням

виведення відсортованого одновимірного масиву

**Кінець**

*Крок 4.*

**Початок**

k = 0

flag = true

A[5][5]

B[5]

**повторити**

**для** і **від** 0 **до** 5 **із кроком** 1

**повторити**

**для** j **від** 0 **до** 5 **із кроком** 1

**Введення A[i][j]**

**все повторити**

**все повторити**

**повторити**

**для** і **від** 0 **до** 5 **із кроком** 1

**якщо** A[i][i] > 0

**то**

B[k] = A[i][i]

k = k + 1

**все якщо**

**все повторити**

сортування одновимірного масиву бульбашкою за спаданням

виведення відсортованого одновимірного масиву

**Кінець**

*Крок 5.*

**Початок**

k = 0

flag = true

A[5][5]

B[5]

**повторити**

**для** і **від** 0 **до** 5 **із кроком** 1

**повторити**

**для** j **від** 0 **до** 5 **із кроком** 1

**Введення A[i][j]**

**все повторити**

**все повторити**

**повторити**

**для** і **від** 0 **до** 5 **із кроком** 1

**якщо** A[i][i] > 0

**то**

B[k] = A[i][i]

k = k + 1

**все якщо**

**все повторити**

**поки** flag

**то**

**повторити**

**для** і **від** 0 **до** k-1 **із кроком** 1

flag = false

**якщо** B[i] < B[i + 1]

**то**

el = B[i]

B[i] = B[i + 1]

B[i + 1] = el

flag = true

**все якщо**

**все повторити**

**все повторити**

виведення відсортованого одновимірного масиву

**Кінець**

*Крок 6.*

**Початок**

k = 0

flag = true

A[5][5]

B[5]

**повторити**

**для** і **від** 0 **до** 5 **із кроком** 1

**повторити**

**для** j **від** 0 **до** 5 **із кроком** 1

**Введення A[i][j]**

**все повторити**

**все повторити**

**повторити**

**для** і **від** 0 **до** 5 **із кроком** 1

**якщо** A[i][i] > 0

**то**

B[k] = A[i][i]

k = k + 1

**все якщо**

**все повторити**

**поки** flag

**то**

**повторити**

**для** і **від** 0 **до** k-1 **із кроком** 1

flag = false

**якщо** B[i] < B[i + 1]

**то**

el = B[i]

B[i] = B[i + 1]

B[i + 1] = el

flag = true

**все якщо**

**все повторити**

**все повторити**

**повторити**

**для** і **від** 0 **до** k **із кроком** 1

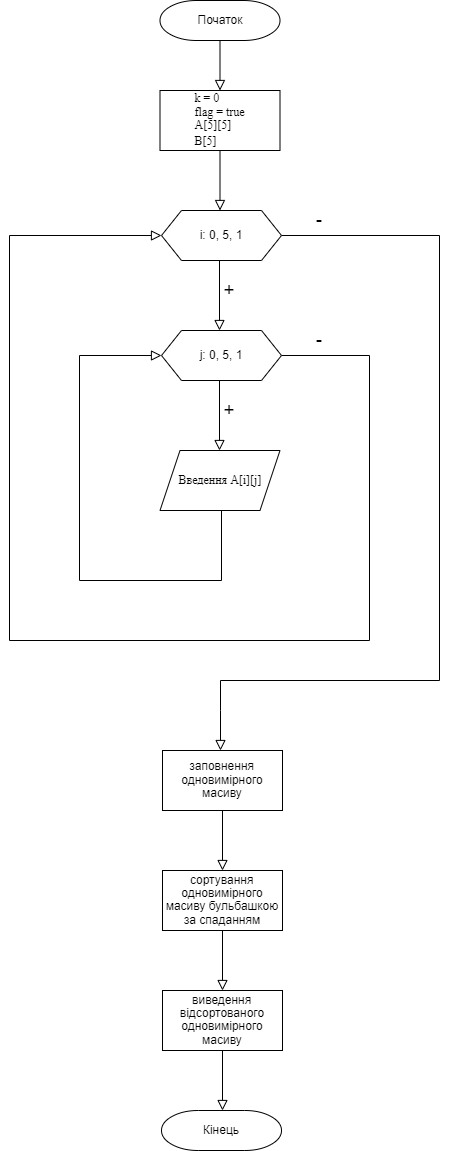
**Виведення B[i]**

**все повторити**

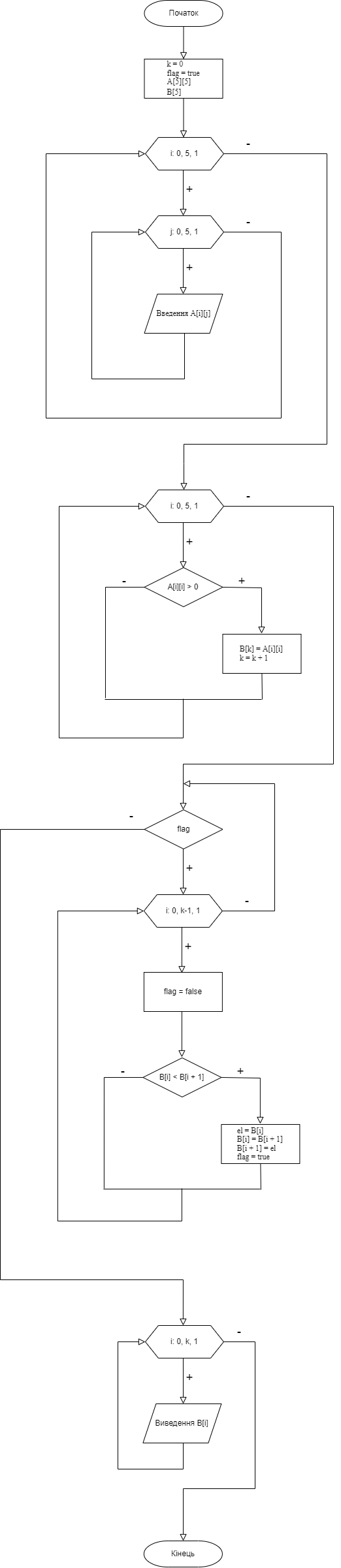
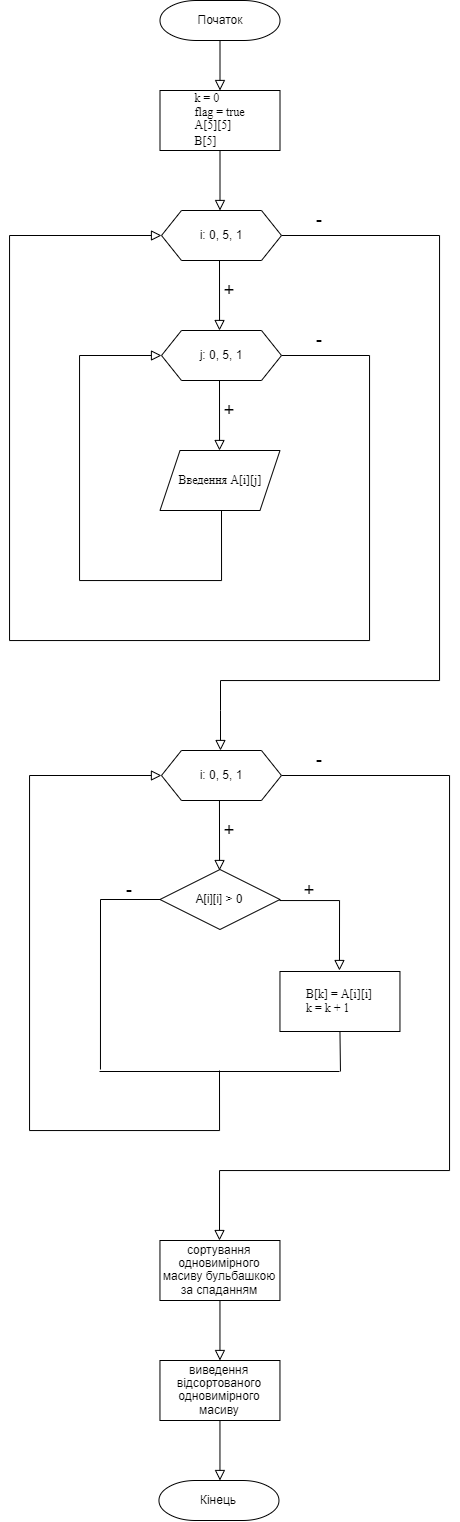
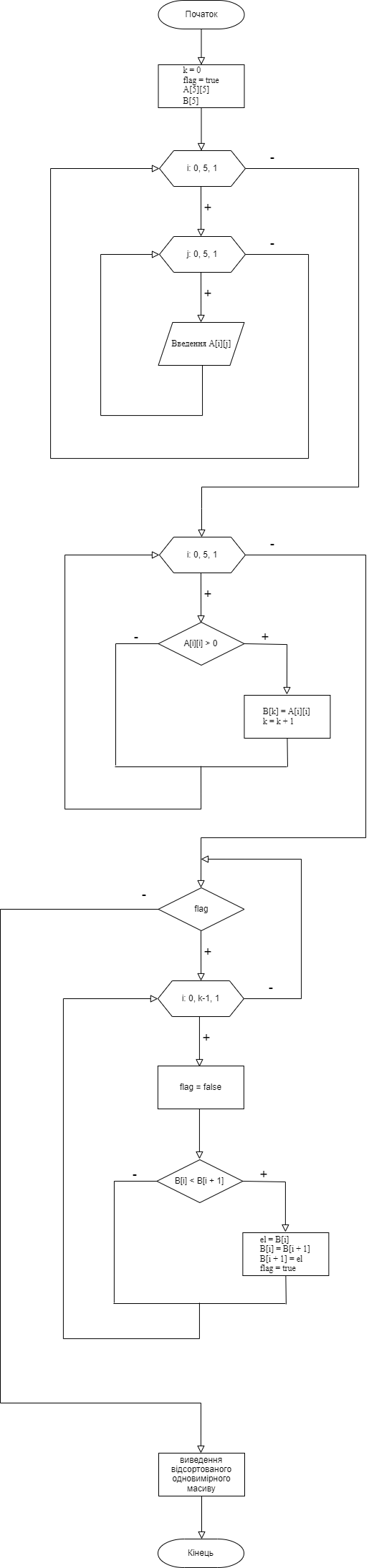
**Кінець**

***Блок-схема.***

*Крок 1. Крок 2. Крок 3.*



*Крок 4. Крок 5.*



*Крок 6.*

***Код програми на мові С++***

#include <iostream>

#include <math.h>

#include <stdio.h>

#include <ctime>

#include <iomanip>

using namespace std;

int main() {

srand(time(NULL));

int\*\* A = new int\* [5];

for (int i = 0; i < 5; i++) A[i] = new int[5];

int\* B = new int[5];

int k = 0;

bool flag = true;

for (int i = 0; i < 5; i++) {

for (int j = 0; j < 5; j++) {

A[i][j] = (rand() % 1000) \* pow(-1, rand());

}

}

cout << "Your matrix:" << endl;

for (int i = 0; i < 5; i++) {

for (int j = 0; j < 5; j++) {

cout << setw(5) << A[i][j];

}

cout << endl;

}

cout << endl;

for (int i = 0; i < 5; i++) {

if (A[i][i] > 0) {

B[k] = A[i][i];

k++;

}

}

cout << "Array of positive elements of the main diagonal of the matrix: ";

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

cout << setw(5) << B[i];

}

cout << endl;

int el;

while (flag) {

for (int i = 0; i < k - 1; i++) {

flag = false;

if (B[i] < B[i + 1]) {

el = B[i];

B[i] = B[i + 1];

B[i + 1] = el;

flag = true;

}

}

}

cout << "Sorted array: ";

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

cout << setw(5) << B[i];

}

cout << endl;

for (int i = 0; i < 5; i++) {

delete[] A[i];

}

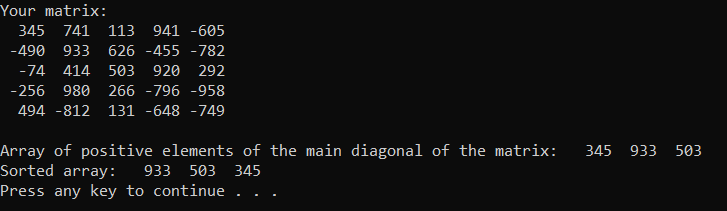
delete[] A;

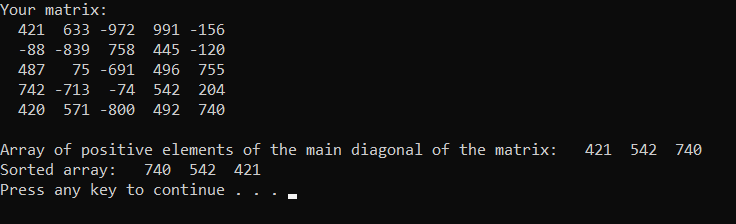
delete[] B;

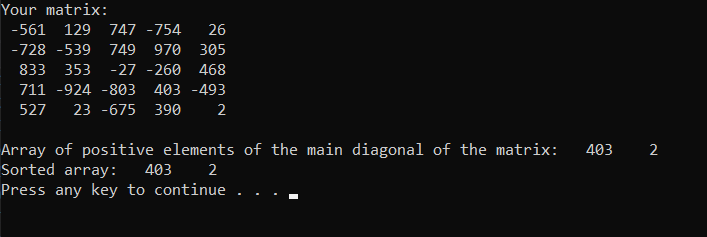
system("pause");

return 0;

}







**IV. Висновки.**

Було досліджено підходи до пошуку та перетворення на матрицях та було набуто практичних навичок використання укладених керувальних дій повторення і їх з’єднання під час складання програмних специфікацій, застосувавши алгоритм пошуку для знаходження всіх додатних елементів головної діагоналі заданої квадратної матриці та алгоритм сортування обміном (бульбашкою) за спаданням в одновимірному масиві, заповненому цими додатними елементами головної діагоналі.