Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний

інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної инженериії

Звіт

з лабораторної работи №8 з дисципліни

«Алгоритми та структури даних-1.

Основи алгоритмізації»

«Дослідження алгоритмів пошуку та сортування»

Варіант 1

Виконав студент ІП-14 Аджигельдієва Мадіна Алімівна

Перевірів Мартинова Оксана Петрівна

Київ 2021

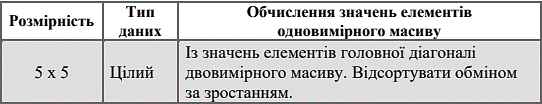
**Лабораторна робота 8**

**Дослідження алгоритмів пошуку та сортування**

**Мета –** дослідити алгоритми пошуку та сортування, набути практичних навичок використання цих алгоритмів під час складання програмних специфікацій.

Задача 1. Розробити алгоритм та написати програму, яка складається з наступних дій:

1. Опису змінної індексованого типу (двовимірний масив).
2. Ініціювання змінної, що описана в п.1 даного завдання.
3. Створення нової змінної індексованого типу (одновимірний масив) та її ініціювання значеннями.



**Розв’язання**

**Постановка задачі.** Результатом розв’язку є виведення одновимірного масива, який відсортований обміном за зростанням.

**Побудова математичної моделі.**

|  |  |  |
| --- | --- | --- |
| **Змінна** | **Тип** | **Призначення** |
| SIZE | Цілий | Постійна змінна для визначення розміру масива |
| printArray | – | Функція виведення масива (ФВМ) |
| inputMatrix[SIZE][SIZE] | Цілий | Ім'я параметра ФВМ |
| inputArray[SIZE] |
| i, j | Цілий | Локальні змінні ФВМ |
| inputMatrix[SIZE][SIZE] | Цілий | Двовимірний масив |
| i, j, temp | Цілий | Локальні змінні в основній функції |
| resultArray[SIZE] | Цілий | Результат  одновимірний масив |

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

Визначимо основні дії:

*Крок 1.* Введення локальних змінних в основнії функції та в функціях користувача.

*Крок 2.* Ініціалізація двовимірного масива inputMatrix[SIZE][SIZE] з числами.

*Крок 3.* Виведення двовимірного масива inputMatrix[SIZE][SIZE].

*Крок 4.* Поміщення діагональних елементів матриці inputMatrix[SIZE][SIZE] в resultArray[SIZE].

*Крок 5.* Сортування м. «Бульбашка» елементів resultArray[SIZE] за зростанням.

*Крок 6.* Виведення одновимірного масива resultArray[SIZE].

**Псевдокод алгоритму.**

*Крок 1.*

**початок**

Введення const SIZE, i, j, temp, resultArray[SIZE], inputMatrix[SIZE][SIZE]

Ініціалізація двовимірного масива inputMatrix[SIZE][SIZE]

Виведення двовимірного масива inputMatrix[SIZE][SIZE]

Поміщення діагональних елементів матриці inputMatrix[SIZE][SIZE] в resultArray[SIZE]

Сортування м. «Бульбашка» елементів resultArray[SIZE] за зростанням

Виведення одновимірного масива resultArray[SIZE]

**кінець**

*Крок 2.*

**початок**

Введення const SIZE = 5, i, j, temp, resultArray[SIZE], inputMatrix[SIZE][SIZE]

**для** (i = 0; I < SIZE; i++)

**для** (j = SIZE – 1; j >= 0; j--)

inputMatrix[i][j] = (i + 1) \* (j +(i + 1) \* pow(-1, i);

Виведення двовимірного масива inputMatrix[SIZE][SIZE]

Поміщення діагональних елементів матриці inputMatrix[SIZE][SIZE] в resultArray[SIZE]

Сортування м. «Бульбашка» елементів resultArray[SIZE] за зростанням

Виведення одновимірного масива resultArray[SIZE]

**кінець**

*Крок 3.*

**початок**

Введення const SIZE = 5, i, j, temp, resultArray[SIZE], inputMatrix[SIZE][SIZE]

**для** (i = 0; I < SIZE; i++)

**для** (j = SIZE – 1; j >= 0; j--)

inputMatrix[i][j] = (i + 1) \* (j +(i + 1) \* pow(-1, i);

**printArray**(inputMatrix);

Поміщення діагональних елементів матриці inputMatrix[SIZE][SIZE] в resultArray[SIZE]

Сортування м. «Бульбашка» елементів resultArray[SIZE] за зростанням

Виведення одновимірного масива resultArray[SIZE]

**кінець**

*Крок 4.*

**початок**

Введення const SIZE = 5, i, j, temp, resultArray[SIZE], inputMatrix[SIZE][SIZE]

**для** (i = 0; I < SIZE; i++)

**для** (j = SIZE – 1; j >= 0; j--)

inputMatrix[i][j] = (i + 1) \* (j +(i + 1) \* pow(-1, i);

**printArray**(inputMatrix);

**для** (i = 0; I < SIZE; i++)

j = i;

resultArray[i] = inputMatrix[i][j];

**виведення** resultArray[i];

Сортування м. «Бульбашка» елементів resultArray[SIZE] за зростанням

Виведення одновимірного масива resultArray[SIZE]

**кінець**

*Крок 5.*

**початок**

Введення const SIZE = 5, i, j, temp, resultArray[SIZE], inputMatrix[SIZE][SIZE]

**для** (i = 0; I < SIZE; i++)

**для** (j = SIZE – 1; j >= 0; j--)

inputMatrix[i][j] = (i + 1) \* (j +(i + 1) \* pow(-1, i);

**printArray**(inputMatrix);

**для** (i = 0; I < SIZE; i++)

j = i;

resultArray[i] = inputMatrix[i][j];

**виведення** resultArray[i];

**для** (i = 0; I < SIZE - 1; i++)

**для** (j = 0; j < SIZE – 1; j++)

**якщо** (resultArray[j] > resultArray[j + 1])

temp = resultArray[j];

resultArray[j] = resultArray[j + 1];

resultArray[j + 1] = temp;

Виведення одновимірного масива resultArray[SIZE]

**кінець**

*Крок 6.*

**початок**

Введення const SIZE = 5, i, j, temp, resultArray[SIZE], inputMatrix[SIZE][SIZE]

**для** (i = 0; I < SIZE; i++)

**для** (j = SIZE – 1; j >= 0; j--)

inputMatrix[i][j] = (i + 1) \* (j +(i + 1) \* pow(-1, i);

**printArray**(inputMatrix);

**для** (i = 0; I < SIZE; i++)

j = i;

resultArray[i] = inputMatrix[i][j];

**виведення** resultArray[i];

**для** (i = 0; I < SIZE - 1; i++)

**для** (j = 0; j < SIZE – 1; j++)

**якщо** (resultArray[j] > resultArray[j + 1])

temp = resultArray[j];

resultArray[j] = resultArray[j + 1];

resultArray[j + 1] = temp;

**printArray**(resultArray);

**кінець**

**Псевдокод функцій.**

*Крок 1.*

**початок**

Введення i, j

Ініціювання inputMatrix[SIZE][SIZE]

**кінець**

*Крок 2.*

**початок**

Введення i, j

Ініціювання inputMatrix[SIZE][SIZE]

**кінець**

*Крок 3.*

**початок**

Введення i, j

**для** (i = 0; i < SIZE; i++)

**для** (j = 0; j < SIZE; j++)

**виведення** inputMatrix[i][j]

**кінець**

*Крок 1.*

**початок**

Введення i

Ініціювання inputArray[SIZE]

**кінець**

*Крок 2.*

**початок**

Введення i

Ініціювання inputArray[SIZE]

**кінець**

*Крок 3.*

**початок**

Введення i, j

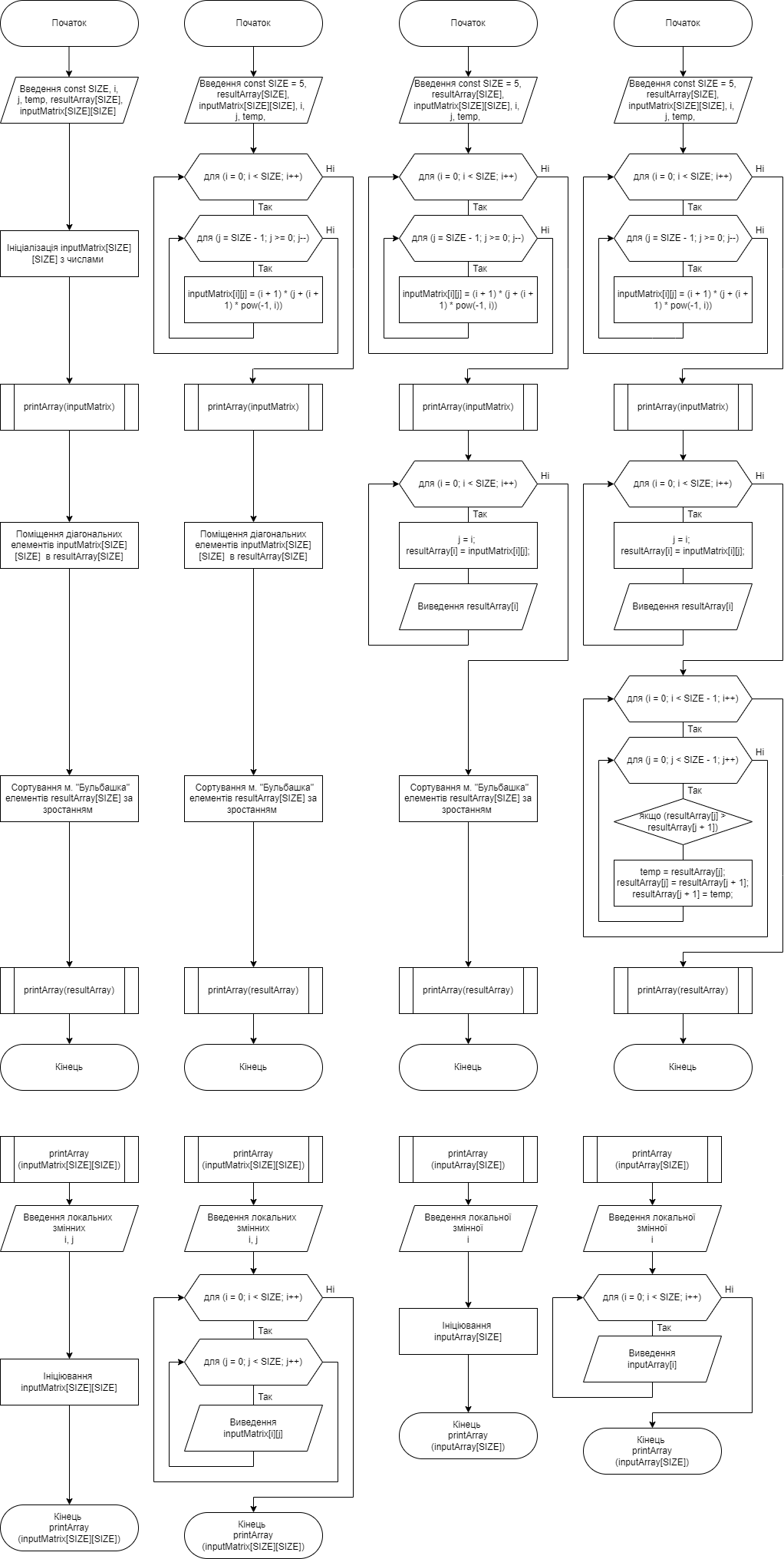
**для** (i = 0; i < SIZE; i++)

**виведення** inputMatrix[i][j]

**кінець**

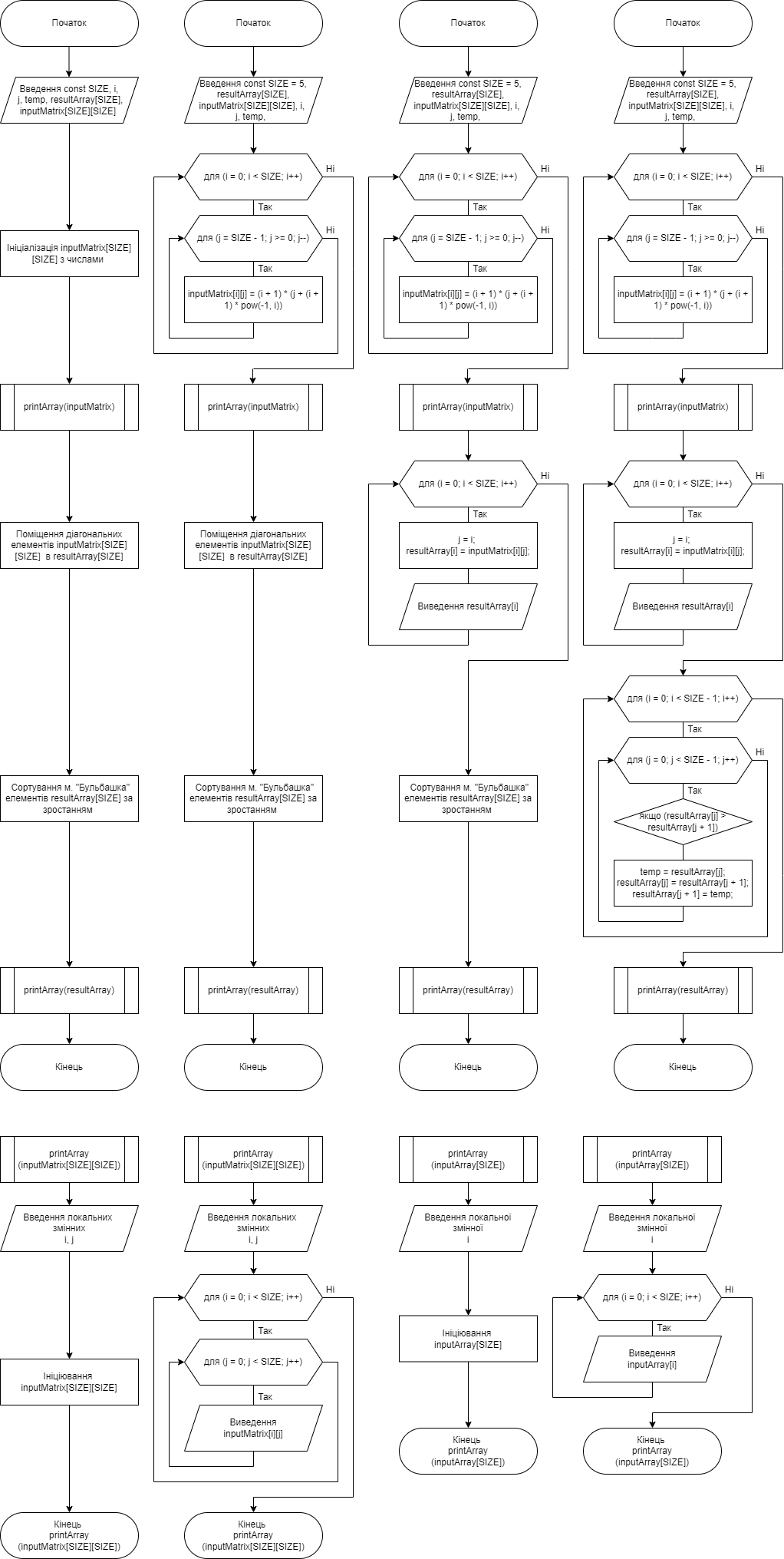
**Блок-схема алгоритму.**

*Крок 1 Крок 2 Крок 3 Крок 4*



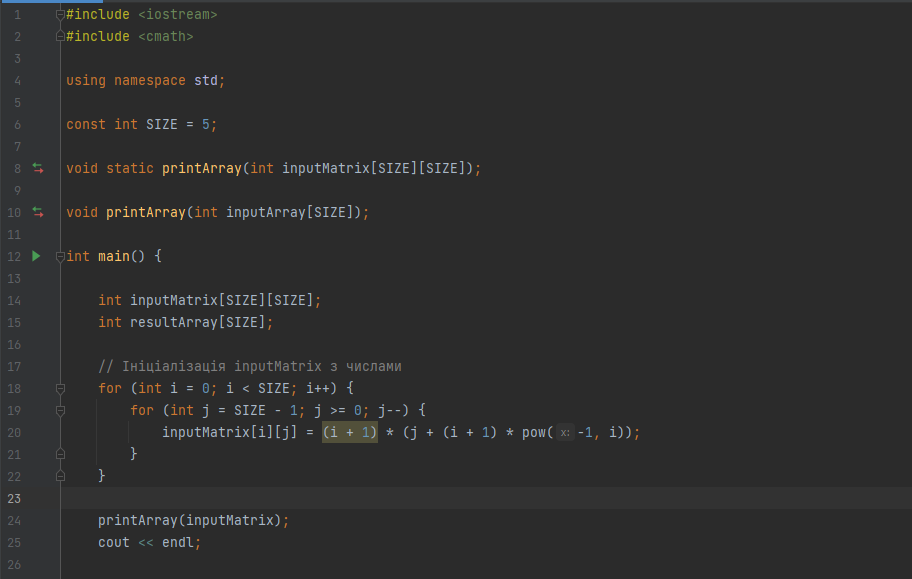
**Блок-схема функцій.**

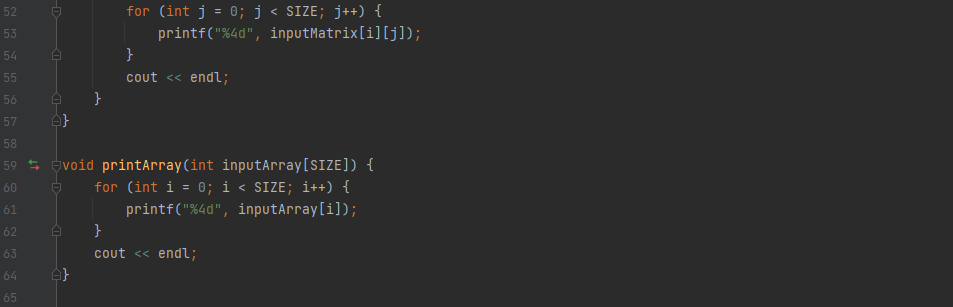
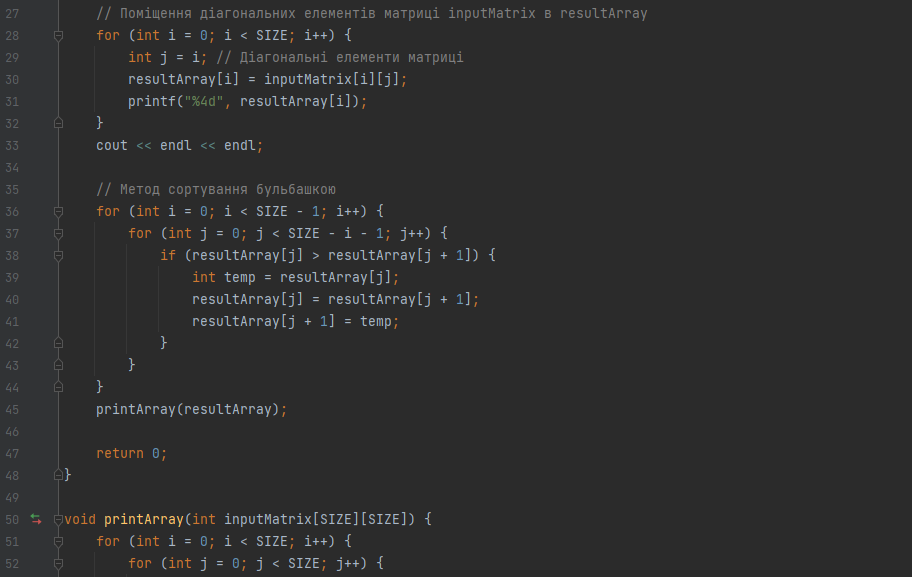
*Крок1 Крок 2 Крок 1 Крок2*



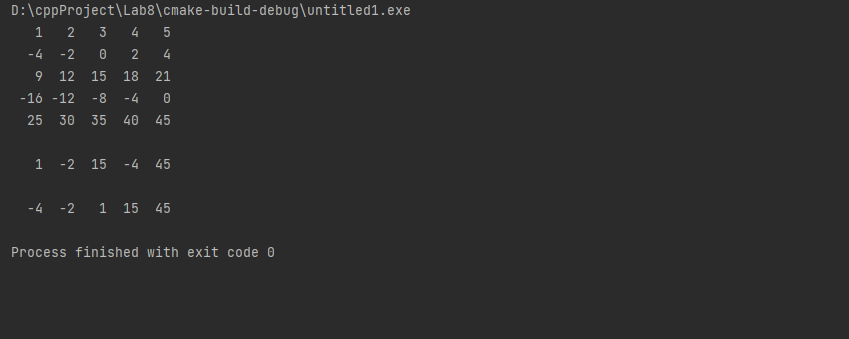
**Код програми.**

Представимо у вигляді коду програми та виведемо результат роботи.





**Результат коду програми.**



**Випробування алгоритму.**

Перевіримо правільність алгоритму на довільних конкретних значеннях початкових даних.

|  |  |
| --- | --- |
| **Блок** | **Дія** |
|  | **Початок** |
| 1 | **Введення** const SIZE = 5, resultArray[SIZE],  inputMatrix[SIZE][SIZE], i, j, temp |
| 2 | **для** (i = 0; i < SIZE; i++)  **для** (j = SIZE - 1; j >= 0; j--)  inputMatrix[i][j] = (i + 1) \* (j + (i + 1) \* pow(-1, i)); |
| 3 | printArray(inputMatrix);  1 2 3 4 5  -4 -2 0 2 4  9 12 15 18 21  -16 -12 -8 -4 0  25 30 35 40 45 |
| 4 | **для** (i = 0; i < SIZE; i++)  j = i;  resultArray[i] = inputMatrix[i][j];  **виведення** resultArray;  1 -2 15 -4 45 |
| 5 | **для** (i = 0; i < SIZE - 1; i++)  **для** (j = 0; j < SIZE - 1; j++)  **якщо** (resultArray[j] > resultArray[j + 1])  temp = resultArrat[j];  resultArray[j] = resultArray[j + 1]  resultArray[j + 1] = temp; |
| 6 | printArray(resultArray);  -4 -2 1 15 45 |
|  | **Кінець** |

**Випробування функцій.**

|  |  |
| --- | --- |
| **Блок** | **Дія** |
|  | **Початок** |
| 1 | **Введення** i, j |
| 2 | **для** (i = 0; i < SIZE; i++)  для (j = 0; j < SIZE; j++)  **виведення** inputMatrix[i][j] |
|  | **Кінець** |

|  |  |
| --- | --- |
| **Блок** | **Дія** |
|  | **Початок** |
| 1 | **Введення** i |
| 2 | **для** (i = 0; i < SIZE; i++)  **виведення** inputArray[i] |
|  | **Кінець** |

**Висновок.**

В результаті виконання лабораторної роботи:

- набула теоритичні знання о алгоритмах пошуку та методах сортування: класифікації алгоритмів пошуку, властивості та види алгоритмів сортування, основні групи алгоритмів сортування.

- визначила основні кроки дій, записала програмні специфікації у псевдокоді, зобразила алгоритм у графічній формі у вигляді блок-схеми.

- представила програмні спеціфікації у вигляді коду програми на мові програмування С++, отримала правильний результат, виконала перевірку алгоритму підставляючи конкретні значення у певні дані.