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 № 9 з дисципліни

«Алгоритми та структури даних-1.

Основи алгоритмізації»

«ДСЛІДЖЕННЯ АЛГОРИТМІВ ОБХОДУ МАСИВІВ»

Варіант 2

Виконав студент ІП-14, Бабіч Денис Володимирович (шифр, прізвище, ім'я, по батькові)

Перевірила Мартинова Оксана Петрівна

( прізвище, ім'я, по батькові)

Київ 2021

**Мета:** дослідити алгоритми обходу масивів, набути практичних навичок використання цих алгоритмів під час складання програмних специфікацій.

**Варіант 2**

**Задача**:

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

1. Опису змінної індексованого типу (двовимірний масив) згідно з варіантом.

2. Ініціювання змінної, що описана в п.1 даного завдання.

3. Обчислення змінної, що описана в п.1, згідно з варіантом.



**Постановка задачі:**

Для розв’язку завдання потрібний двомірний масив , size – кількість рядків і стовпців (за умовою задачі, матриця - квадратична), проходом по рядам «змійкою». У ході розв’язання завдання будуть використаний оператор встановлення подільності числа для встановлення з кінця чи початку рядка потрібно проходити матрицю. Якщо індекс – парне число, то з початку. Якщо ні – то з кінця. На першій ітерації і дорівнюватиме 0, число – парне, тому обхід починається з початку. Коли і = 1, то обхід починається з кінця. Також використовуватиметься стандартна функція для генерування випадкового числа у обраному діапазоні (-100, 100). На етапі підрахунку суми використовуватиметься змінна **index** для того щоб додавати елементи починаючи з певного номеру, оскілки потрібно сумувати елементи лише ті, які знаходяться над головною діагоналю, то на першій ітерації **index** = 1 (сумуються елементи починаючи з першого за індексом), на другій ітерації **index** = 2 (сумуються елементи починаючи з другого за індексом). Всі елементи, які підлягають сумуванню на кожній ітерації сумуються із змінною **sum**.

**Побудова математичної моделі:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Змінна** | **Тип** | **Ім’я** | **Призначення** |
| Матриця | Дійсний  Двовимірний масив | matrix | Матриця для для заповнення випадковими числами |
| Індекс | Ціле число | index | Індекс, за яким знаходиться число у матриці, яке треба просумувати |
| Зміння для збереження числа | Дійсний | sum | Підрахунок суми елементів матриці, які задовольняють умову |
| Кількість рядків і стовпців квадратичної матриці | Ціле число | size | Кількість рядків і стовпців квадратичної матриці |
| Ітератор у арифметичному циклі | Ціле число | i | Ітератор у циклі |
| Ітератор у арифметичному циклі | Ціле число | j | Ітератор у циклі |

**Розв’язання**

Крок 1. Введення початкових даних.

Крок 2. Ініціалізація початкових даних

Крок 3.Деталізувати заповнення матриці.

Крок 4. Деталізувати пошук суми елементів над головною діагоналлю.

Крок 5. Деталізувати виведення результату.

**Псевдокод**

|  |  |
| --- | --- |
| Крок 1 | **Початок**  Введення початкових даних.  Ініціалізація початкових даних.  Деталізувати заповнення матриці.  Деталізувати пошук суми елементів над головною діагоналлю.  Деталізувати виведення результату.  **Кінець** |
| Крок 2 | **Початок**  Введення **size**.  Ініціалізація початкових даних.  Деталізувати заповнення матриці.  Деталізувати пошук суми елементів над головною діагоналлю.  Деталізувати виведення результату.  **Кінець** |
| Крок 3 | **Початок**  Введення **size**.  sum := 0  index := 1  Деталізувати заповнення матриці.  Деталізувати пошук суми елементів над головною діагоналлю.  Деталізувати виведення результату.  **Кінець** |
| Крок 4 | **Початок**  Введення **size**.  sum := 0  index := 1  matrix[size][size]  повторити  для i від 0 до size, 1  якщо i % 2 == 0  то  повторити  для j від 0 до size, 1  matrix[i][j] := rand()  інакше  повторити  для j від size – 1 до 0, -1  matrix[i][j] := rand()  Деталізувати пошук суми елементів над головною діагоналлю.  Деталізувати виведення результату.  **Кінець** |
| Крок 5 | **Початок**  Введення **size**.  sum := 0  index := 1  matrix[size][size]  повторити  для i від 0 до size, 1  якщо i % 2 == 0  то  повторити  для j від 0 до size, 1  matrix[i][j] := rand()  інакше  повторити  для j від size – 1 до 0, -1  matrix[i][j] := rand()  все повторити  все якщо  все повторити  повторити  для i від 0 до size, 1  повторити  для j від index до size, 1  sum := sum + matrix[i][j]  все повторити  index := index + 1  все повторити  Деталізувати виведення результату.  **Кінець** |
| Крок 6 | **Початок**  Введення **size**.  sum := 0  index := 1  matrix[size][size]  повторити  для i від 0 до size, 1  якщо i % 2 == 0  то  повторити  для j від 0 до size, 1  matrix[i][j] := rand()  інакше  повторити  для j від size – 1 до 0, -1  matrix[i][j] := rand()  все повторити  все якщо  все повторити  повторити  для i від 0 до size, 1  повторити  для j від index до size, 1  sum := sum + matrix[i][j]  все повторити  index := index + 1  все повторити  Вивід **sum**.  **Кінець** |

**Блок-схема алгоритму**

**Крок 1**

****

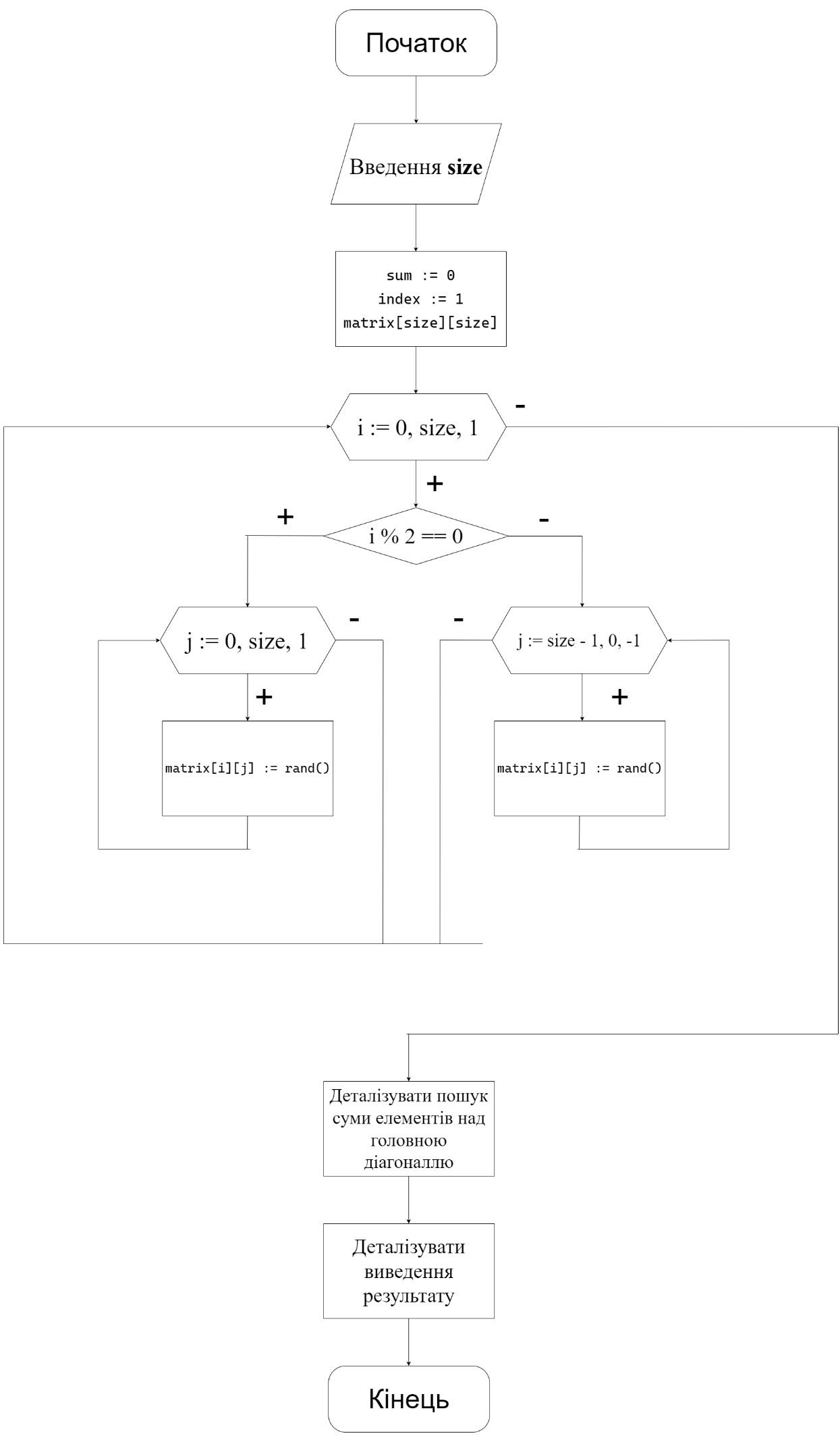
**Крок 2**

****

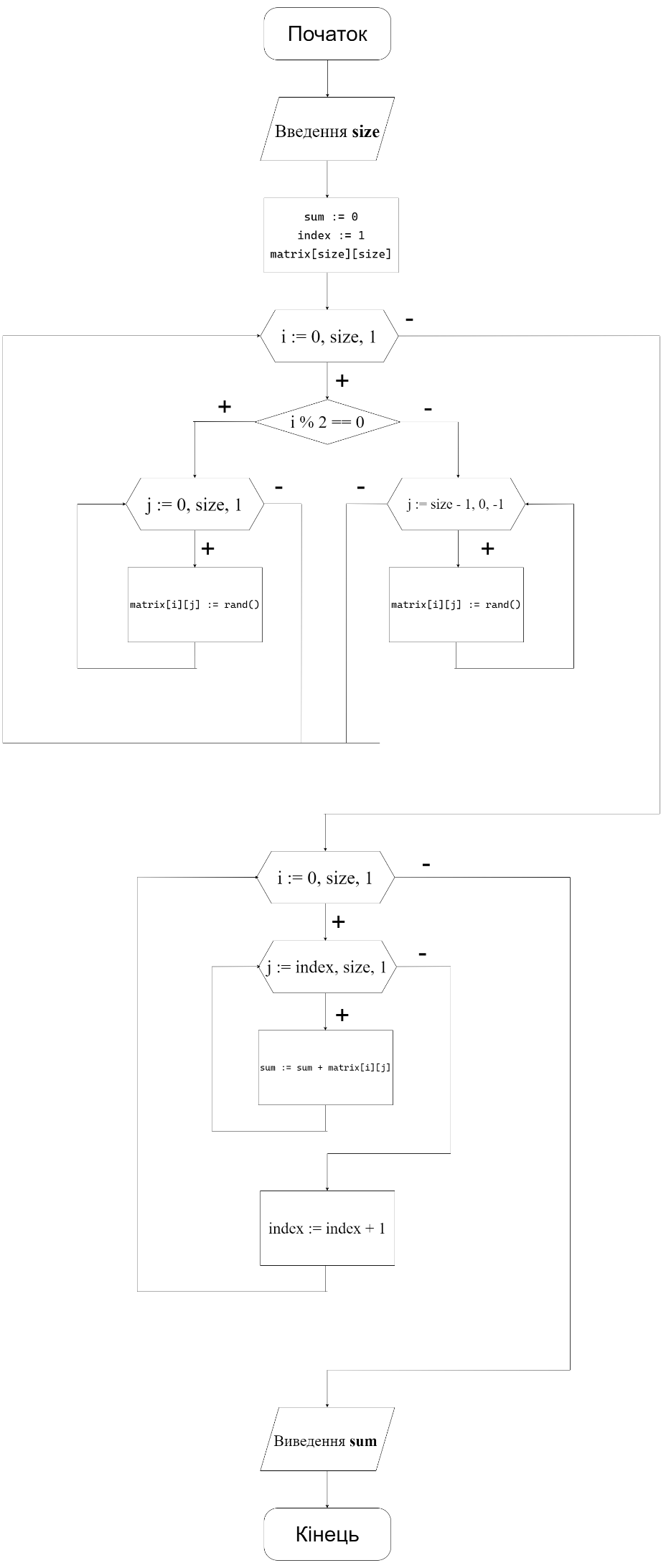
**Крок 3**

****

**Крок 4**

****

**Крок 5**

****

**Випробування алгоритму:**

**Код алгоритму**





**Випробування**

|  |  |
| --- | --- |
| Блок | Дія |
|  | **початок** |
| 1. | //Введення **size**.  size = 5 |
| 2. | //Отримання матриці за допомогою генерування випадкових чисел  //повторити для і від 0 до size //на першій ітерації і % 2 == 0  //тому прохід по матриці починається з початку рядка  // повторити для і від 0 до size //на другій ітерації і % 2 != 0  //тому прохід по матриці починається з кінця рядка рядка  //Сама отримана матриця  61.3147 67.449 -26.8899 -74.3828 34.489  85.992 -68.8772 -70.4093 81.2983 -16.0131  46.8307 76.5862 -15.2623 -34.8247 -82.342  -11.9175 76.3298 52.2874 -56.5477 53.325  94.8424 -32.4686 -86.5902 55.2721 -22.9591 |
| 3. | 1. i = 0, j = 1  sum = sum + matrix[0][1] //0 + 67.449  2. i = 0, j = 2  sum = sum + matrix[0][2] //67.449 + (-26.8899)  .  .  .  i = 4, j = 4  sum = sum + matrix[4][4] //-45.3413 + (-22.9591) |
|  | Виведення: -68.3004 |
|  | **кінець** |
|  |  |

****

**Висновок:**

У результаті лабораторної роботи набув навичок у використані двовимірних масивів, під час виконання роботи використовував оператори альтернативного вибору, арифметичні операції, арифметичні цикли та функцію для створення випадкових чисел у заданому діапазоні. Дослідив прохід двовимірного масиву “змійкою”.