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 обходу масивів»
Варіант 2

Виконав студент ІП-12, Басараб Олег Андрійович
(шифр, прізвище, ім'я, по батькові)

Перевірив _____
(прізвище, ім'я, по батькові)

Київ 2021

Лабораторна робота № 8 “Дослідження алгоритмів обходу масивів”

Варіант 2

Мета – дослідити алгоритми обходів масивів, набути практичних навичок використання цих алгоритмів під час складання програмних специфікацій.

Задача 2. Розробити алгоритм та написати програму, яка складається з наступних дій:

1. Опису змінної індексованого типу (двовимірний масив) згідно з варіантом (матриця дійсних чисел A[n, n]).
2. Ініціювання змінної, що описана в п. 1 даного завдання.
3. Обчислення змінної, що описана в п. 1, згідно з варіантом (сума елементів, розташованих вище головної діагоналі).

Розв'язування.

Постановка задачі. Результатом розв’язку є дійсна змінна sum. Для її знаходження необхідне вхідна ціла змінна n (розмірність двовимірного масиву A[n][n], вважаємо, що n > 0). Початковим даним є двовимірний масив arr2D. Під час розв’язування буде використано функцію rand(), що повертає випадкове дійсне число з проміжку [-100.0; 100.0]. Вважаємо, що масиви передаються до підпрограм за посиланням, а запис float** arr2D означає посилання на дійсний двовимірний масив. Також вважаємо, що заголовок арифметичного циклу виду «для і від 1 до n» означає, що і – лічильник циклу з початковим значенням 1, (i ≤ n) – умова невиходу, а крок = 1.

Побудова математичної моделі. Складемо таблицю імен змінних основної програми.

Змінна	Тип	Ім’я	Призначення
Двовимірний масив	Дійсний двовимірний масив	arr2D [] []	Початкове дане

Сума елементів двовимірного масиву, які вищі головної діагоналі	Дійсний	sum	Результат

Складемо таблицю імен змінних підпрограми getRandomReal2DArray для задання елементів двовимірного масиву arr2D.

Змінна	Тип	Ім’я	Призначення
Кількість рядків двовимірного масиву	Цілий (константа)	ROWS_NUM	Вхідне дане
Кількість стовбців двовимірного масиву	Цілий (константа)	COLUMNS_NUM	Вхідне дане
Лічильник арифметичного циклу	Цілий	i	Поточне дане
Лічильник арифметичного циклу	Цілий	j	Поточне дане
Результуючий двовимірний масив, елементами якого є випадкові дійсні числа	Дійсний двовимірний масив	arr2D [] []	Результат

Складемо таблицю імен змінних підпрограми arrayProcessing для знаходження sum.

Змінна	Тип	Ім’я	Призначення
Масив, елементи якого буду сортуватися	Дійсний масив	arr2D []	Вхідне дане
Розмір масиву arr2D[N][N]	Цілий (константа)	N	Вхідне дане

Лічильник арифметичного циклу	Цілий	i	Поточне дане
Лічильник арифметичного циклу	Цілий	j	Поточне дане
Сума елементів двовимірного масиву, які вищі головної діагоналі	Дійсний	sum	Результат

Таким чином, формулювання завдання зводить до:

1. **Опису підпрограми float** getRandomReal2DArray(const int ROWS_NUM, const int COLUMNS_NUM) для задання елементів двовимірного масиву arr2D.** В системі з двох арифметичних циклів (для i від 1 до ROWS_NUM – заголовок зовнішнього циклу, для j від 1 до COLUMNS_NUM – заголовок внутрішнього циклу), в тілі внутрішнього циклу, знаходимо елемент arr2D[i][j] = rand(). Підпрограма повертає двовимірний масив arr2D.
2. **Опису підпрограми float arrayProcessing(float** arr2D, const int N) для знаходження sum.** Змінна sum ініціалізується 0. В системі з двох арифметичних циклів (для i від 1 до N – заголовок зовнішнього циклу, для j від 1 до N – заголовок внутрішнього циклу), в тілі внутрішнього циклу, sum += arr2D[i][j].
3. **Опису основного алгоритму.** Вводимо значення N. Задаємо двовимірний масив arr2D за допомогою підпрограми getRandomReal2DArray. Обробляємо arr2D за допомогою arrayProcessing і знаходимо sum. Виводимо sum.

Розіб’ємо алгоритм роботи основної програми на кроки:

Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо дію задання елементів масиву arr2D за допомогою власної підпрограми.
Крок 3. Деталізуємо дію обробки елементів масиву arr2D за допомогою arrayProcessing.

Розіб’ємо алгоритм роботи підпрограми float** getRandomReal2DArray(const int ROWS_NUM, const int COLUMNS_NUM) для задання елементів двовимірного масиву arr2D:
Крок 1. Визначимо основні дії.
Крок 2. Деталізуємо дію знаходження елементів arr2D за допомогою системи двох арифметичних циклів.

Розіб’ємо алгоритм роботи підпрограми float arrayProcessing(float** arr2D, const int N) для знаходження sum:
Крок 1. Визначимо основні дії.
Крок 2. Деталізуємо дію знаходження sum за допомогою системи вкладених циклів.

Програмні специфікації зобразимо у псевдокоді та в графічній формі в блок-схемі алгоритму.

Псевдокод.

Основна програма:

Крок 1

початок

ввід n

задання елементів масиву arr2D за допомогою власної підпрограми

обробка елементів масиву arr2D за допомогою arrayProcessing

вивід sum

кінець

Крок 2

початок

ввід n

arr2D = getRandomReal2DArray(n, n)

обробка елементів масиву arr2D за допомогою arrayProcessing

вивід sum

кінець

Крок 3

початок

ввід n

arr2D = getRandomReal2DArray(n, n)

sum = arrayProcessing(arr2D, n)

вивід sum

кінець

Підпрограма float getRandomReal2DArray(const int ROWS_NUM, const int COLUMNS_NUM):**

Крок 1

початок

знаходження елементів arr2D за допомогою системи двох арифметичних циклів

повернути arr2D

кінець

Крок 2

початок

повторити

для i **від** 1 **до** ROWS_NUM

повторити

для j **від** 1 **до** COLUMNS_NUM

arr2D[i][j] = rand()

все повторити

все повторити

повернути arr2D

кінець

Підпрограма float arrayProcessing(float arr2D, const int N):**

Крок 1

початок

знаходження sum за допомогою системи вкладених циклів

повернути sum

кінець

Крок 2

початок

sum = 0

повторити

для i **від** 1 **до** N

повторити

для j **від** i + 1 **до** N

sum += arr2D[i][j]

все повторити

все повторити

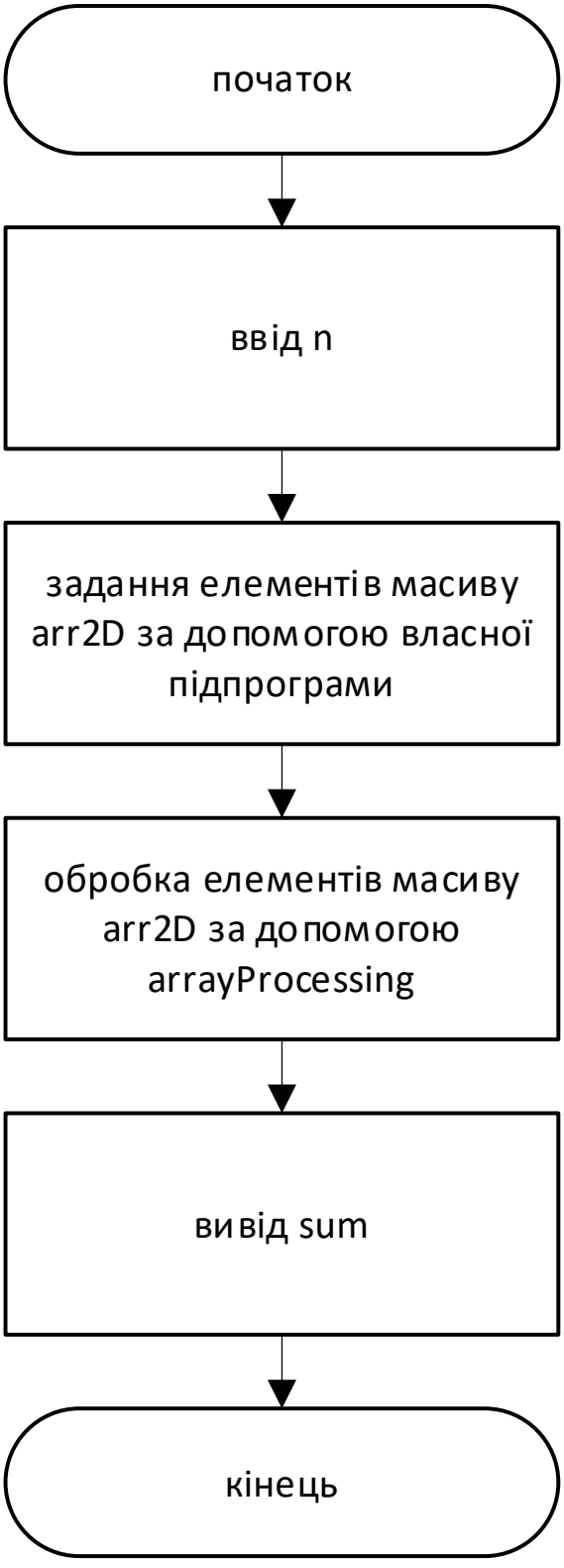
повернути sum

кінець

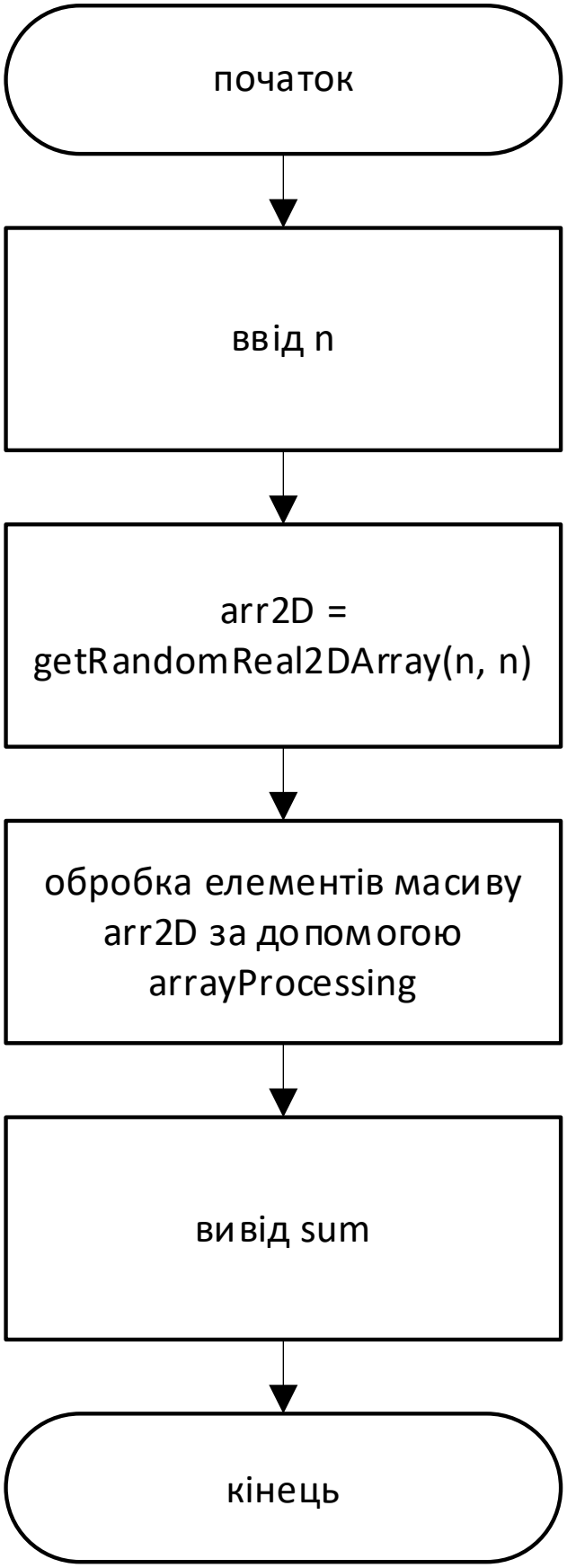
Блок-схема алгоритму.

Основна програма:

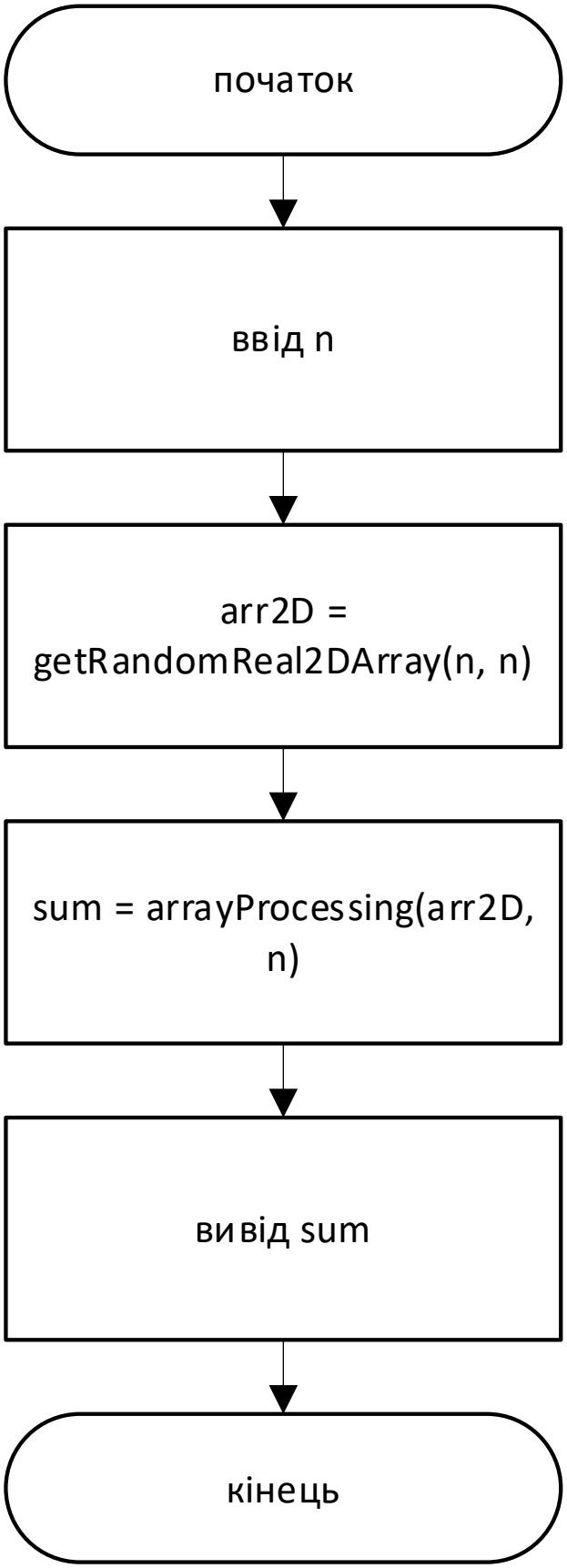
Крок 1



Крок 2

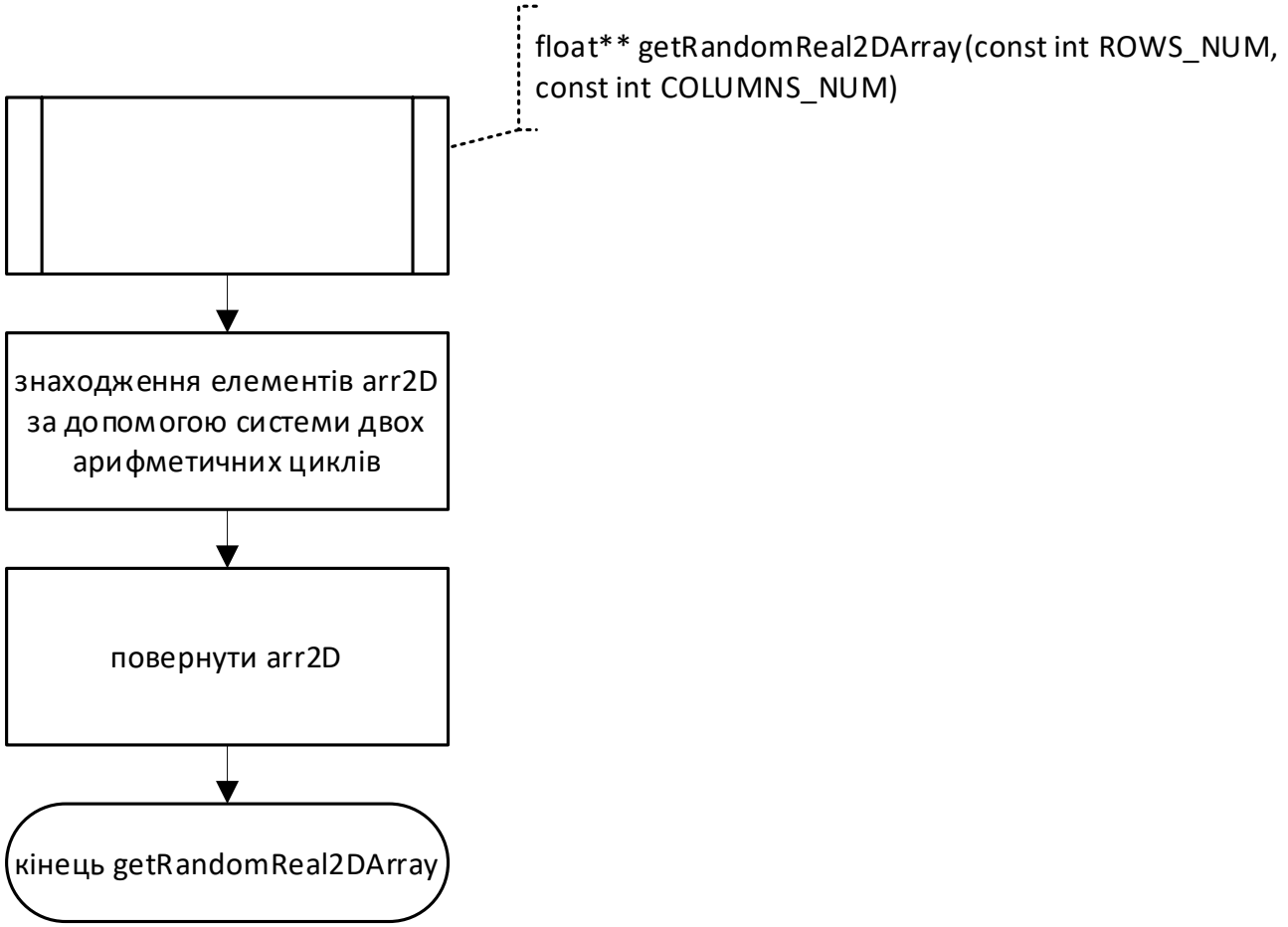

початок
ввід n
arr2D =
getRandomReal2DArray(n, n)
обробка елементів масиву
arr2D за допомогою
arrayProcessing
вивід sum
кінець

Крок 3

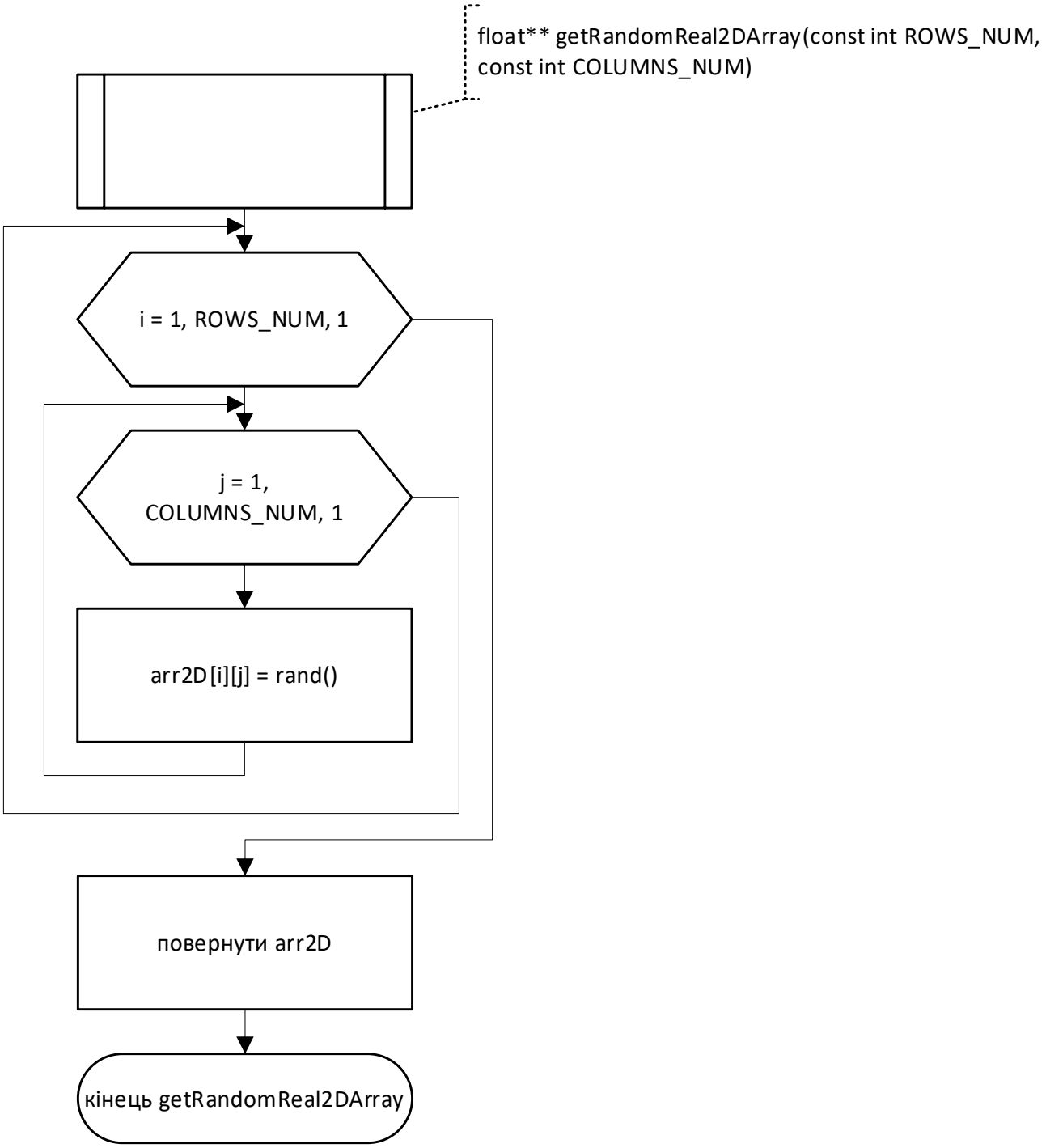

початок
ввід n
arr2D = getRandomReal2DArray(n, n)
sum = arrayProcessing(arr2D, n)
вивід sum
кінець

Підпрограма float getRandomReal2DArray(const int ROWS_NUM, const int COLUMNS_NUM):**

Крок 1



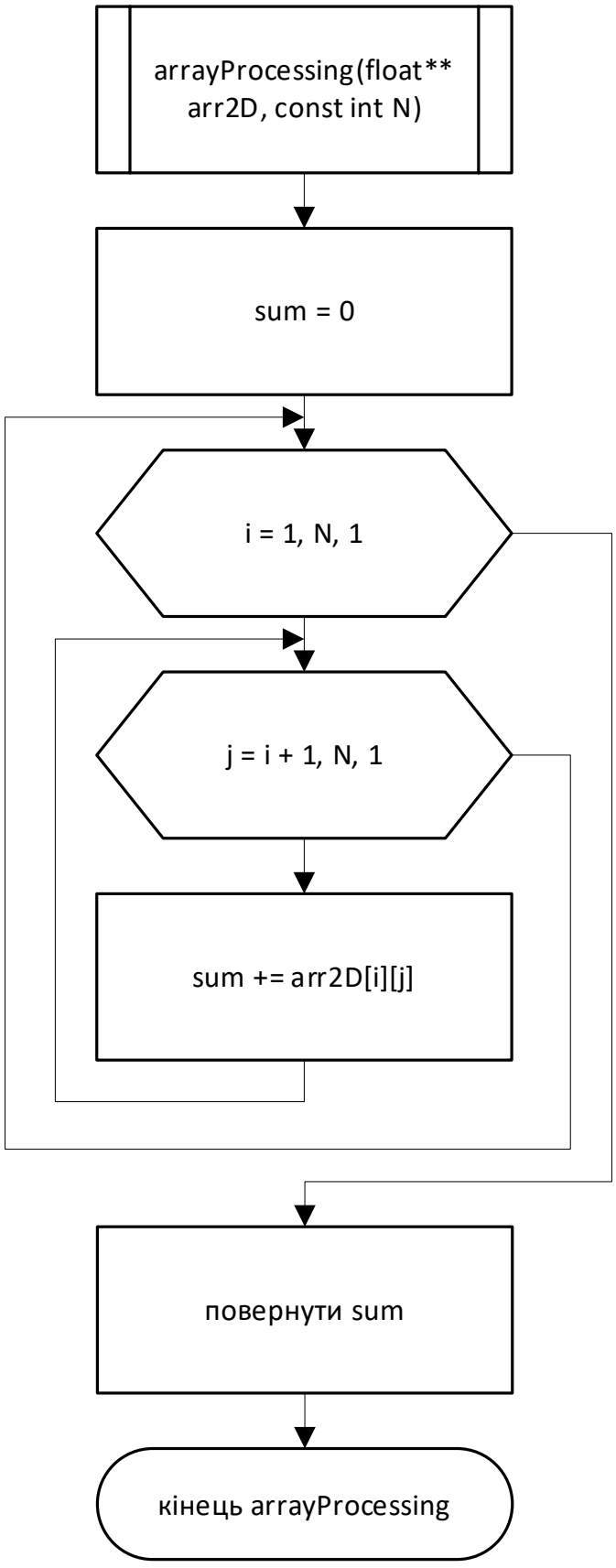
Крок 2


float** getRandomReal2DArray(const int ROWS_NUM,
const int COLUMNS_NUM)
i = 1, ROWS_NUM, 1
j = 1,
COLUMNS_NUM, 1
arr2D[i][j] = rand()
повернути arr2D
кінець getRandomReal2DArray

Підпрограма float arrayProcessing(float arr2D, const int N):**

Крок 1

Крок 2


arrayProcessing(float** arr2D, const int N)
sum = 0
i = 1, N, 1
j = i + 1, N, 1
sum += arr2D[i][j]
повернути sum
кінець arrayProcessing

Реалізація алгоритму на мові C++:

```
#include <iostream>
#include <ctime>
#include <cstdlib>
#include <iomanip>

float getRandomRealNum();

float** getRandomReal2DArray(const size_t ROWS_NUM, const size_t COLUMNS_NUM);

void array2DOutput(float** arr2D, const size_t ROWS_NUM, const size_t COLUMNS_NUM);

void deleteArray2D(float** arr2D, const size_t ROWS_NUM);

float arrayProcessing(float** arr2D, const size_t N);

int main()
{
    float** arr2D;
    int n{};
    do {
        std::cout << "Enter int n (n > 0): ";
        std::cin >> n;
        if (n <= 0) {
            std::cout << "Error: input n isn't correct!\n";
        }
    } while (n <= 0);
    arr2D = getRandomReal2DArray(n, n);
    array2DOutput(arr2D, n, n);

    float sum{};
    sum = arrayProcessing(arr2D, n);
    std::cout << "\nSum is: " << sum << "\n";
    deleteArray2D(arr2D, n);
    return 0;
}

float getRandomRealNum()
{
    float randomRealNum;
    const float MY_RAND_MAX = 100.0, MY_RAND_MIN = -100.0;
    randomRealNum = MY_RAND_MIN + static_cast <float> (rand()) / (static_cast <float>
(RAND_MAX / (MY_RAND_MAX - MY_RAND_MIN)));
    return randomRealNum;
}

float** getRandomReal2DArray(const size_t ROWS_NUM, const size_t COLUMNS_NUM)
{
    srand(static_cast <unsigned int> (time(NULL)));

    float** arr2D = new float* [ROWS_NUM];

    for (size_t i{ 0 }; i < ROWS_NUM; ++i) {
        arr2D[i] = new float[COLUMNS_NUM];
    }

    for (size_t i{ 0 }; i < ROWS_NUM; ++i) {
        for (size_t j{ 0 }; j < COLUMNS_NUM; ++j) {
            arr2D[i][j] = getRandomRealNum();
```

```
        }
    }
    return arr2D;
}

void array2DOutput(float** arr2D, const size_t ROWS_NUM, const size_t COLUMNS_NUM)
{
    std::cout << "\nElements of two-dimensional array:\n";
    for (size_t i{ 0 }; i < ROWS_NUM; ++i) {
        for (size_t j{ 0 }; j < COLUMNS_NUM; ++j) {
            std::cout << std::setw(15) << arr2D[i][j];
        }
        std::cout << "\n";
    }
}

void deleteArray2D(float** arr2D, const size_t ROWS_NUM)
{
    for (size_t i{ 0 }; i < ROWS_NUM; ++i) {
        delete[] arr2D[i];
    }
    delete[] arr2D;
}

float arrayProcessing(float** arr2D, const size_t N)
{
    float sum{};

    for (size_t i{ 0 }; i < N; ++i) {
        for (size_t j = i + 1; j < N; ++j) {
            sum += arr2D[i][j];
        }
    }
    return sum;
}
```

Тестування програми:

sum = -93.4812 - 4.87991 - 98.0651 = -196.426

Висновки. Таким чином, в результаті виконання лабораторної роботи було досліджено алгоритми обходів масивів (на прикладі обходу елементів квадратної матриці, що знаходяться вище головної діагоналі) та набуто практичних навичок використання цих алгоритмів під час складання програмних специфікацій. Особливістю виконання цього варіанту лабораторної є написання кількох власних підпрограм та задання елементів двовимірного масиву за допомогою функції rand().