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»

«Дослідження алгоритмів обходу масивів»

Варіант 32

Виконав студент ІП-14, Шляхтун Денис Михайлович

(шифр, прізвище, ім'я, по батькові)

Перевірив доц. кафедри ІПІ Мартинова Оксана Петрівна

( прізвище, ім'я, по батькові)

Київ 2021

**Лабораторна робота 9**

**Дослідження алгоритмів обходу масивів**

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

**Задача**: Задано матрицю дійсних чисел A[m,n]. При обході матриці змійкою по стовбцям знайти в ній перший додатний елемент Х. Порівняти Х з середньоарифметичним значенням елементів над побічною діагоналлю

**Постановка задачі**. Результатом розв’язку є порівняння елементу Х з середньоарифметичним значенням елементів над побічною діагоналлю матриці. Х – перше додатне число, що знаходиться обходом матриці змійкою по стовбцям. Початкові дані – матриця, що генерується випадковим чином, та розмірність матриці, що вводиться користувачем.

**Побудова математичної моделі**. Складемо таблицю імен змінних

|  |  |  |  |
| --- | --- | --- | --- |
| ***Змінна*** | ***Тип*** | ***Ім’я*** | ***Призначення*** |
| *Основні змінні* | | | |
| Кількість рядків | Цілий | m | Початкове дане |
| Кількість стовбців | Цілий | n | Початкове дане |
| Двовимірний масив | Дійсний | arr[m][n] | Початкове дане |
| Середнє значення | Дійсний | av | Результат |
| Перший додатній елемент | Дійсний | X | Результат |
| *Змінні, що використовуються у підпрограмах* | | | |
| Лічильники | Цілий | i, k | Проміжне значення |
| Сума елементів над діагоналлю | Дійсний | sum | Проміжне значення |
| К-сть елементів над діагоналлю | Цілий | num | Проміжне значення |

Враховуючи те, що для виконання алгоритму потрібна діагональ матриці, сама матриця повинна бути квадратною, отже m і n співпадають.

Для обходу матриці змійкою будемо знаходити остачу від ділення лічильника на 2, з парним значенням прохід відбувається вниз по стовпчику, з непарним значенням – вгору.

Для виконання алгоритму були використані наступні функції зі стандартних бібліотек:

* srand(time(NULL)) та rand() – генерація випадкових чисел для масиву

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

Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо дію генерації масиву.

Крок 3. Деталізуємо дію визначення середнього значення елементів над побічною діагоналлю.

Крок 4. Деталізуємо дію визначення першого додатного елемента обходом змійкою по стовбцям.

Крок 5. Деталізуємо дію порівняння додатного елемента та середнього значення.

**Псевдокод.**

Крок 1.

**початок**

генерація першого масиву

знаходження середнього значення елементів

знаходження першого додатного числа

порівняння середнього значення та першого додатного числа

**кінець**

Крок 2.

**початок**

arr = arrayRand(arr, m, n)

знаходження середнього значення елементів

знаходження першого додатного числа

порівняння середнього значення та першого додатного числа

**кінець**

Крок 3.

**початок**

arr = arrayRand(arr, m, n)

av = average(arr, m, n)

знаходження першого додатного числа

порівняння середнього значення та першого додатного числа

**кінець**

Крок 4.

**початок**

arr = arrayRand(arr, m, n)

av = average(arr, m, n)

X = findX(arr, m, n)

порівняння середнього значення та першого додатного числа

**кінець**

Крок 5.

**початок**

arr = arrayRand(arr, m, n)

av = average(arr, m, n)

X = findX(arr, m, n)

check(X, av)

**кінець**

**функція** arrayRand(arr[][], m, n)

**повторити для** i = 0; i < m; i++

**повторити для** k = 0; k < n; k++

arr[i][k] = ( rand() % 199 – 99 ) / 10

**все повторити**

**все повторити**

**повернути** arr

**кінець функції**

**функція** average(arr[][], m, n)

sum = 0

num = 0

**повторити для** i = 0; i < m - 1; i++

**повторити для** k = 0; k < n – i - 1; k++

sum += arr[i][k]

num++

**все повторити**

**все повторити**

**повернути** sum / num

**кінець функції**

**функція** findX(arr[][], m, n)

**повторити для** i = 0; i < n; i++

**якщо** i % 2 = 0

**повторити для** k = 0; k < m; k++

**якщо** arr[k][i] > 0

**повернути** arr[k][i]

**все якщо**

**все повторити**

**інакше**

**повторити для** k = m - 1; k > -1; k--

**якщо** arr[k][i] > 0

**повернути** arr[k][i]

**все якщо**

**все повторити**

**все повторити**

**кінець функції**

**функція** check(X, av)

**якщо** X > av

**виведення** «X > av»

**інакше якщо** X = av

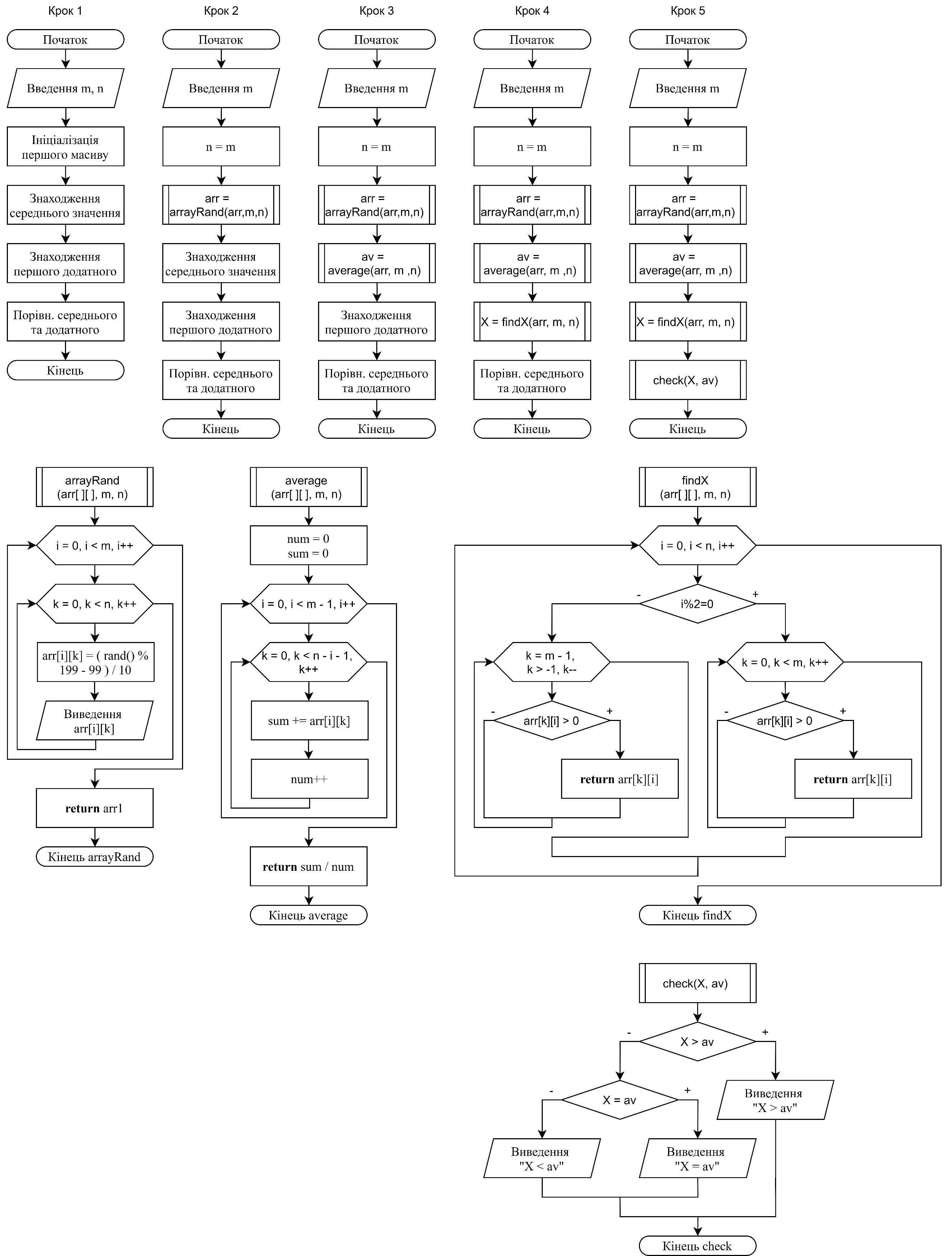
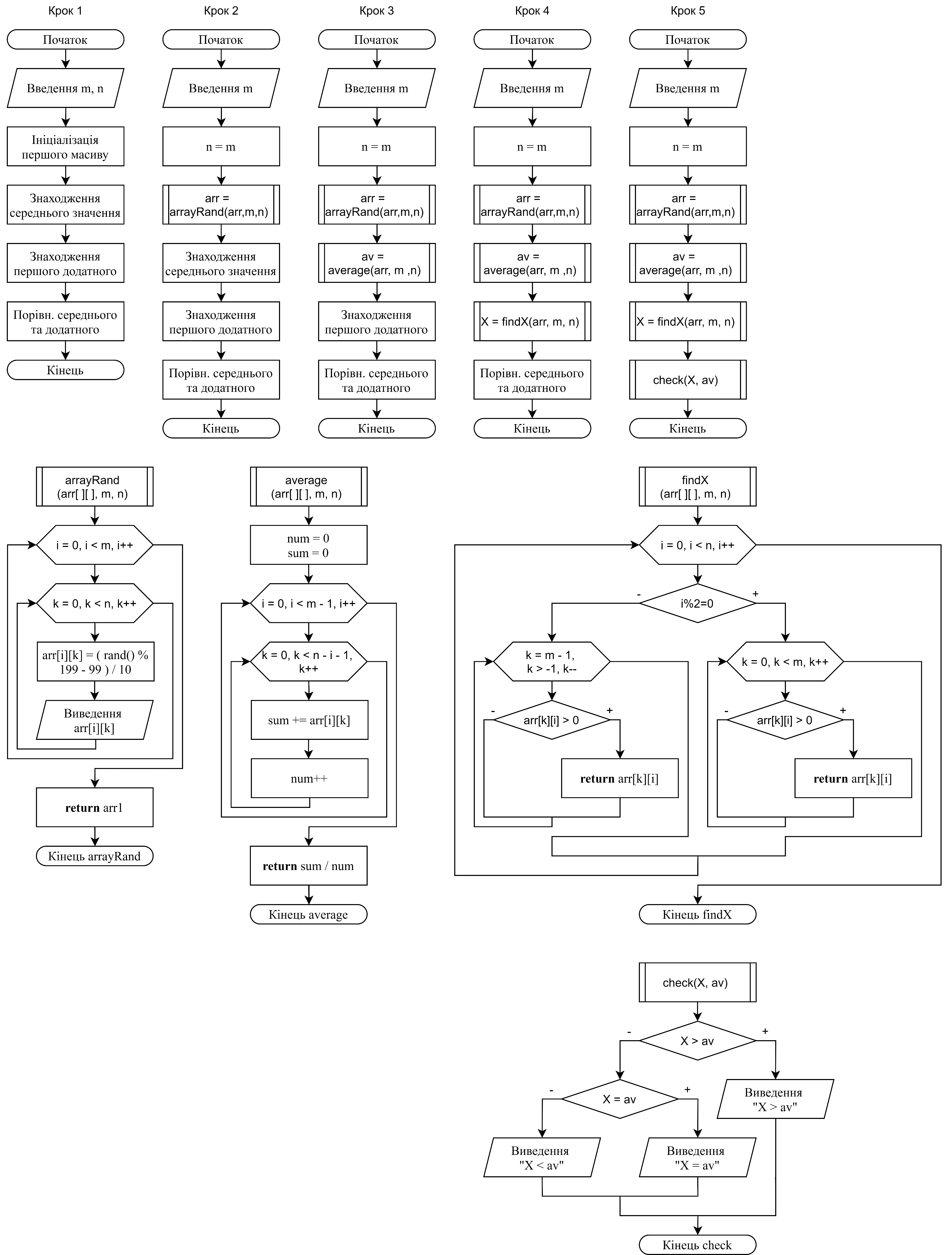
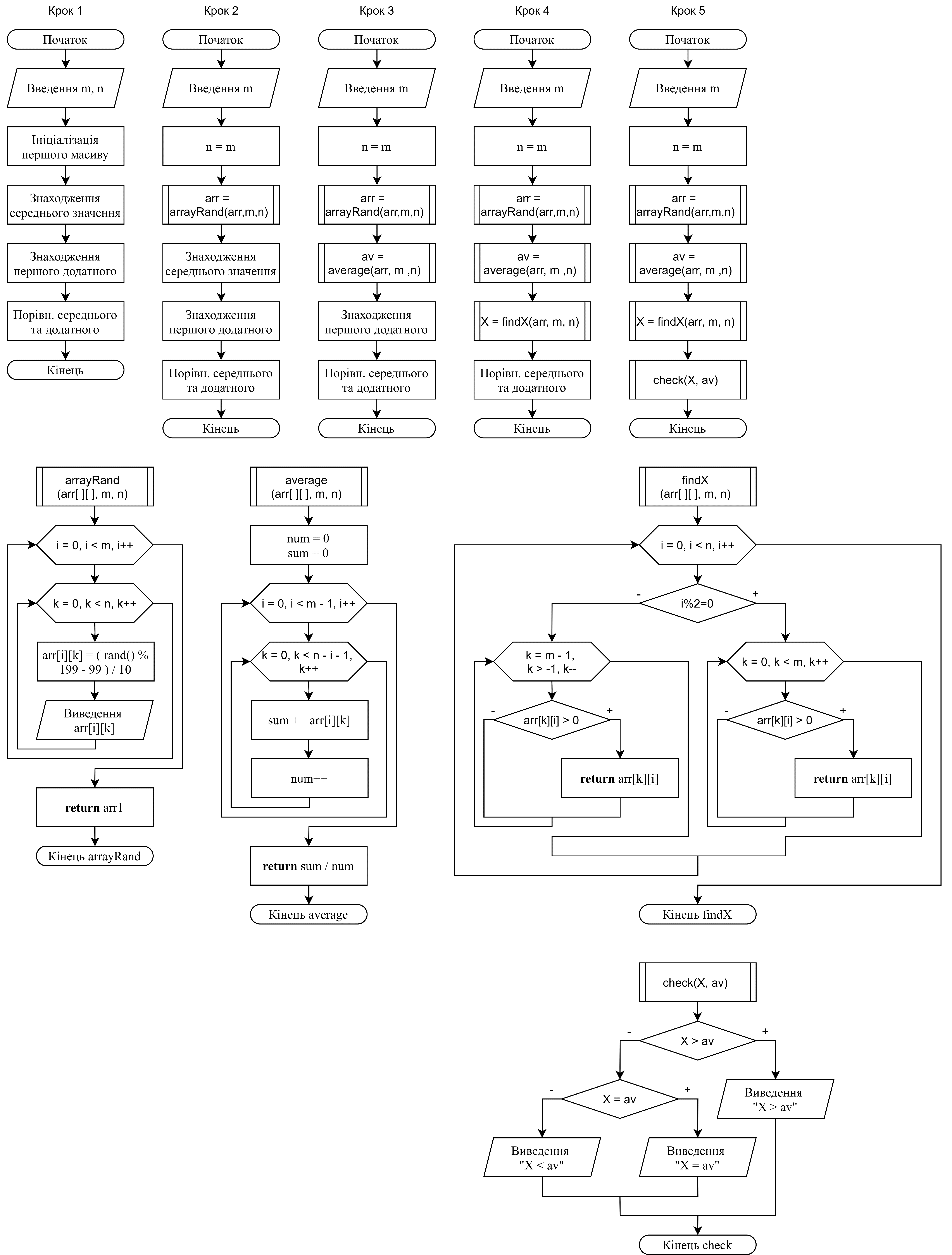
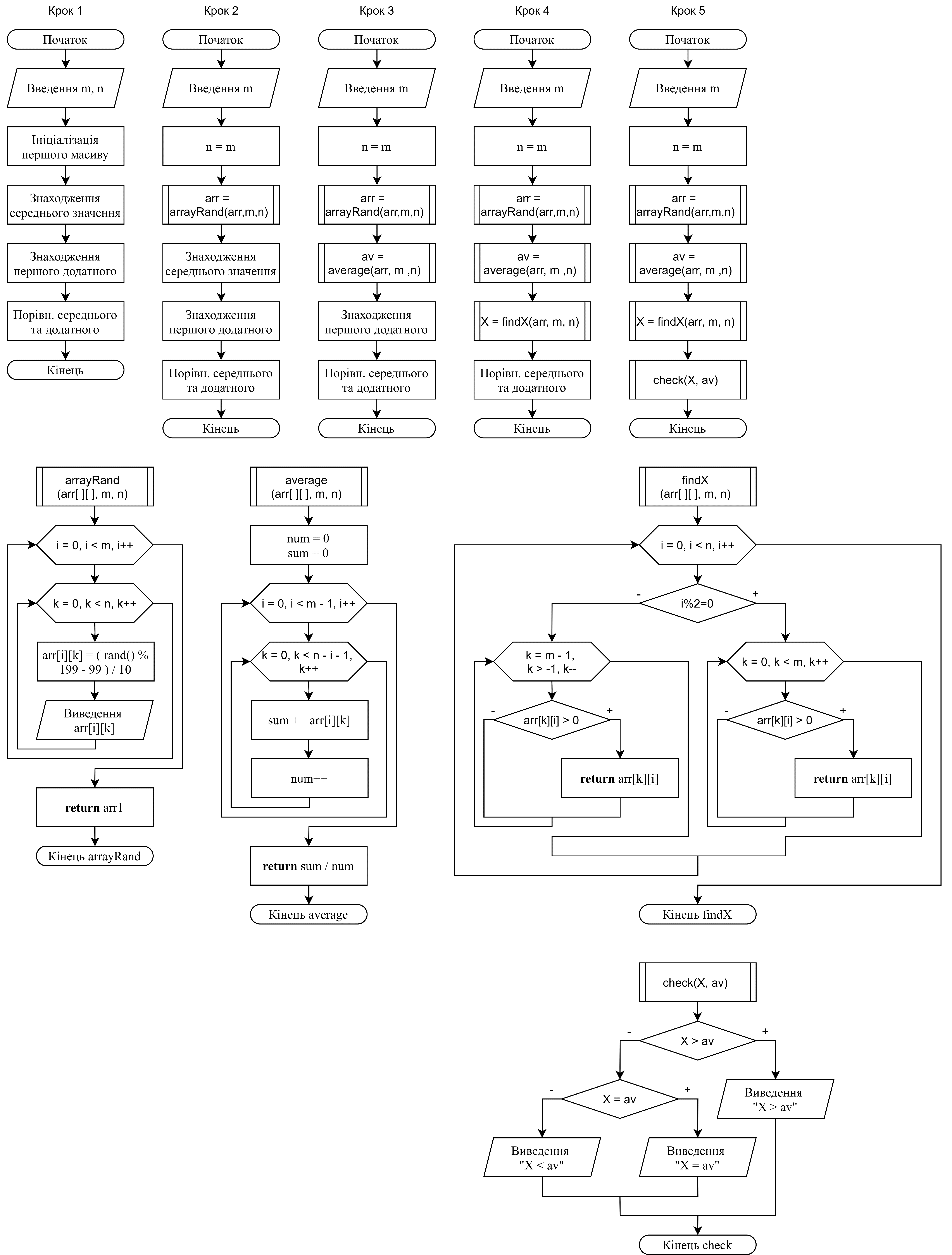
**виведення** «X = av»

**інакше**

**виведення** «X < av»

**кінець функції**

**Блок-схема алгоритму**



**Код програми (C++).**

**Зображення, що містить текст

Автоматично згенерований описЗображення, що містить текст, екран

Автоматично згенерований описЗображення, що містить текст

Автоматично згенерований описЗображення, що містить текст

Автоматично згенерований опис**

**Результат виконання програми.**

**Зображення, що містить текст

Автоматично згенерований опис** **Зображення, що містить текст, електроніка, клавіатура

Автоматично згенерований опис**

**Перевірка виконання.**

Перше виконання:

1. (-6.9+1.2-9.3-8.4-7.4-3.7)/6 = -5.75 – середнє значення визначено правильно
2. Оскільки прохід відбувається змійкою по стовбцям, то Х визначено правильно
3. 1.4 > -5.75 – визначено правильно

Перше виконання:

1. (-9.4-6+5.2+4.8-9.1+5.5-8.1+4.6+1+8.4-7.9+3.8+0.4-3.2-0.6)/15 = -0.7067 – середнє значення визначено правильно
2. Прохід відбувається по стовбцям, Х визначено правильно
3. 5.5 > -0.7 – визначено правильно

**Висновок.**

При виконанні лабораторної роботи було використано матрицю – іменовану сукупність послідовностей значень одного типу, де кожен елемент має два порядкові номери. Для виконання задачі був складений алгоритм для обходу матриці змійкою по стовбцям. Також був досліджений алгоритм пошуку елементів над побічною діагоналлю матриці. В результаті роботи складено математичну модель, покрокові псевдокод та блок-схему, написано програму на мові C++ та успішно перевірено правильність її виконання відносно поставленої задачі.