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 №6 з дисципліни

«Алгоритми та структури даних-1.

Основи алгоритмізації»

«Дослідження алгоритмів пошуку та сортування»

Варіант 33

Виконав студент ІП-12 Чікінеєва Дар’я Олегівна

(шифр, прізвище, ім'я, по батькові)

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

( прізвище, ім'я, по батькові)

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.

**Варіант 33**

**Задача:** Створити цілочисельний двовимірний масив, що складається з 8 рядків та 7 стовпців. Після створити цілочисельний одновимірний масив, елементами якого є добутки додатніх елементів кожного рядка двовимірного масиву.

1. Постановка задачі:

Для вирішення даної задачі нам необхідно створити двовимірний масив, який ми заповнимо рандомними значеннями. Після створення даної матриці ми маємо порахувати добутки всіх її рядків і створити одновимірний масив з цих добутків, який у наступному кроці маємо відсортувати від більного значення до меншого.

1. Побудова математичної моделі:

| Змінна | Тип | Ім’я | Призначення |
| --- | --- | --- | --- |
| являє собою добуток додатніх чисел кожного рядка матриці | Ціле число | dobutok | допоміжна змінна |
| допоміжна змінна для алгоритму сортування | Ціле число | term | допоміжна змінна |
| масив, що зберігає добутки додатніх чисел кожного рядка матриці | одновимірний масив | array\_dob[] | масив значень |
| матриця, на основі якої ми будемо будувати одновимірний масив з добутків додатніх чисел всіх її рядків | Матриця | matrix[][] | матриця значень |

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

Крок 1:

Визначити основні дії

Крок 2:

Створення матриці та виведення її значень

Крок 3:

Знаходження добутку всіх додатніх елементів кожного рядка

Крок 4:

Створення масиву з добутків всіх рядків

Крок 5:

Виведення масиву з добутків всіх рядків

Крок 6:

Сортування обміном за зростанням

Крок 7:

Виведення впорядкованих значень

1. Псевдокод алгоритму

Крок 1

Початок

Створення матриці та виведення її значень

Знаходження добутку всіх додатніх елементів кожного рядка

Створення масиву з добутків всіх рядків

Виведення масиву з добутків всіх рядків

Сортування обміном за зростанням

Виведення впорядкованих значень

Кінець

Крок 2

Початок

**Повторити для i** від i=0 до і=8-1 з кроком 1

**Повторити для j** від j=0 до j=7-1 з кроком 1

matrix[i][j] = rand() % 101 - 50

**Виведення** matrix[i][j]

**Все повторити**

Знаходження добутку всіх додатніх елементів кожного рядка

Створення масиву з добутків всіх рядків

Виведення масиву з добутків всіх рядків

Сортування обміном за зростанням

Виведення впорядкованих значень

Кінець

Крок 3

Початок

**Повторити для i** від i=0 до і=8-1 з кроком 1

**Повторити для j** від j=0 до j=7-1 з кроком 1

matrix[i][j] = rand() % 101 - 50

**Виведення** matrix[i][j]

**Все повторити**

**Повторити для і** від і=0 до і = 8-1 з кроком 1

dobutok = 1

**Повторити для j** від j=0 до j=7-1 з кроком 1

**Якщо** matrix[i][j] > 0

dobutok \*= matrix[i][j]

**все якщо**

**все повторити для j**

**все повторити для і**

Створення масиву з добутків всіх рядків

Виведення масиву з добутків всіх рядків

Сортування обміном за зростанням

Виведення впорядкованих значень

Кінець

Крок 4

Початок

**Повторити для i** від i=0 до і=8-1 з кроком 1

**Повторити для j** від j=0 до j=7-1 з кроком 1

matrix[i][j] = rand() % 101 - 50

**Виведення** matrix[i][j]

**Все повторити**

**Повторити для і** від і=0 до і = 8-1 з кроком 1

dobutok = 1

**Повторити для j** від j=0 до j=7-1 з кроком 1

**Якщо** matrix[i][j] > 0

dobutok \*= matrix[i][j]

**все якщо**

**все повторити для j**

**array\_dob[i] = dobutok**

**все повторити для і**

Виведення масиву з добутків всіх рядків

Сортування обміном за зростанням

Виведення впорядкованих значень

Кінець

Крок 5

Початок

**Повторити для i** від i=0 до і=8-1 з кроком 1

**Повторити для j** від j=0 до j=7-1 з кроком 1

matrix[i][j] = rand() % 101 - 50

**Виведення** matrix[i][j]

**Все повторити**

**Повторити для і** від і=0 до і = 8-1 з кроком 1

dobutok = 1

**Повторити для j** від j=0 до j=7-1 з кроком 1

**Якщо** matrix[i][j] > 0

dobutok \*= matrix[i][j]

**все якщо**

**все повторити для j**

**array\_dob[i] = dobutok**

**Виведення array\_dob[i]**

**все повторити для і**

Сортування обміном за зростанням

Виведення впорядкованих значень

Кінець

Крок 6

Початок

**Повторити для i** від i=0 до і=8-1 з кроком 1

**Повторити для j** від j=0 до j=7-1 з кроком 1

matrix[i][j] = rand() % 101 - 50

**Виведення** matrix[i][j]

**Все повторити**

**Повторити для і** від і=0 до і = 8-1 з кроком 1

dobutok = 1

**Повторити для j** від j=0 до j=7-1 з кроком 1

**Якщо** matrix[i][j] > 0

dobutok \*= matrix[i][j]

**все якщо**

**все повторити для j**

**array\_dob[i] = dobutok**

**Виведення array\_dob[i]**

**все повторити для і**

**Повторити для i** від i=0 до і=8-1 з кроком 1

**Повторити для j** від j=0 до j=7-1 з кроком 1

**Якщо** array\_dob[j] < array\_dob[j + 1]

term = array\_dob[j + 1];

array\_dob[j + 1] = array\_dob[j];

array\_dob[j] = term

**все якщо**

**все повторити для j**

**все повторити для і**

Виведення впорядкованих значень

Кінець

Крок 7

Початок

**Повторити для i** від i=0 до і=8-1 з кроком 1

**Повторити для j** від j=0 до j=7-1 з кроком 1

matrix[i][j] = rand() % 101 - 50

**Виведення** matrix[i][j]

**Все повторити**

**Повторити для і** від і=0 до і = 8-1 з кроком 1

dobutok = 1

**Повторити для j** від j=0 до j=7-1 з кроком 1

**Якщо** matrix[i][j] > 0

dobutok \*= matrix[i][j]

**все якщо**

**все повторити для j**

**array\_dob[i] = dobutok**

**Виведення array\_dob[i]**

**все повторити для і**

**Повторити для i** від i=0 до і=8-1 з кроком 1

**Повторити для j** від j=0 до j=7-1 з кроком 1

**Якщо** array\_dob[j] < array\_dob[j + 1]

term = array\_dob[j + 1];

array\_dob[j + 1] = array\_dob[j];

array\_dob[j] = term

**все якщо**

**все повторити для j**

**все повторити для і**

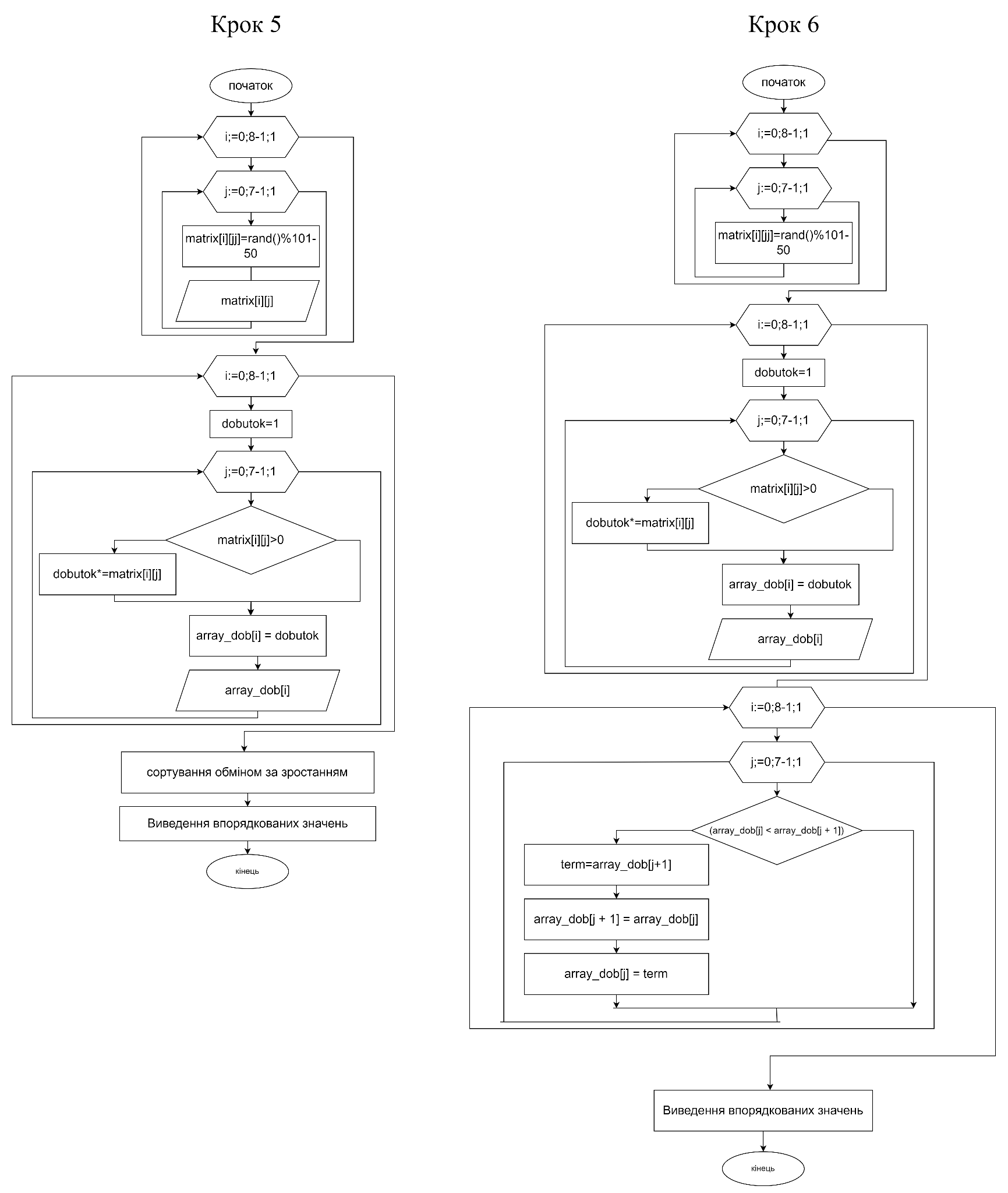
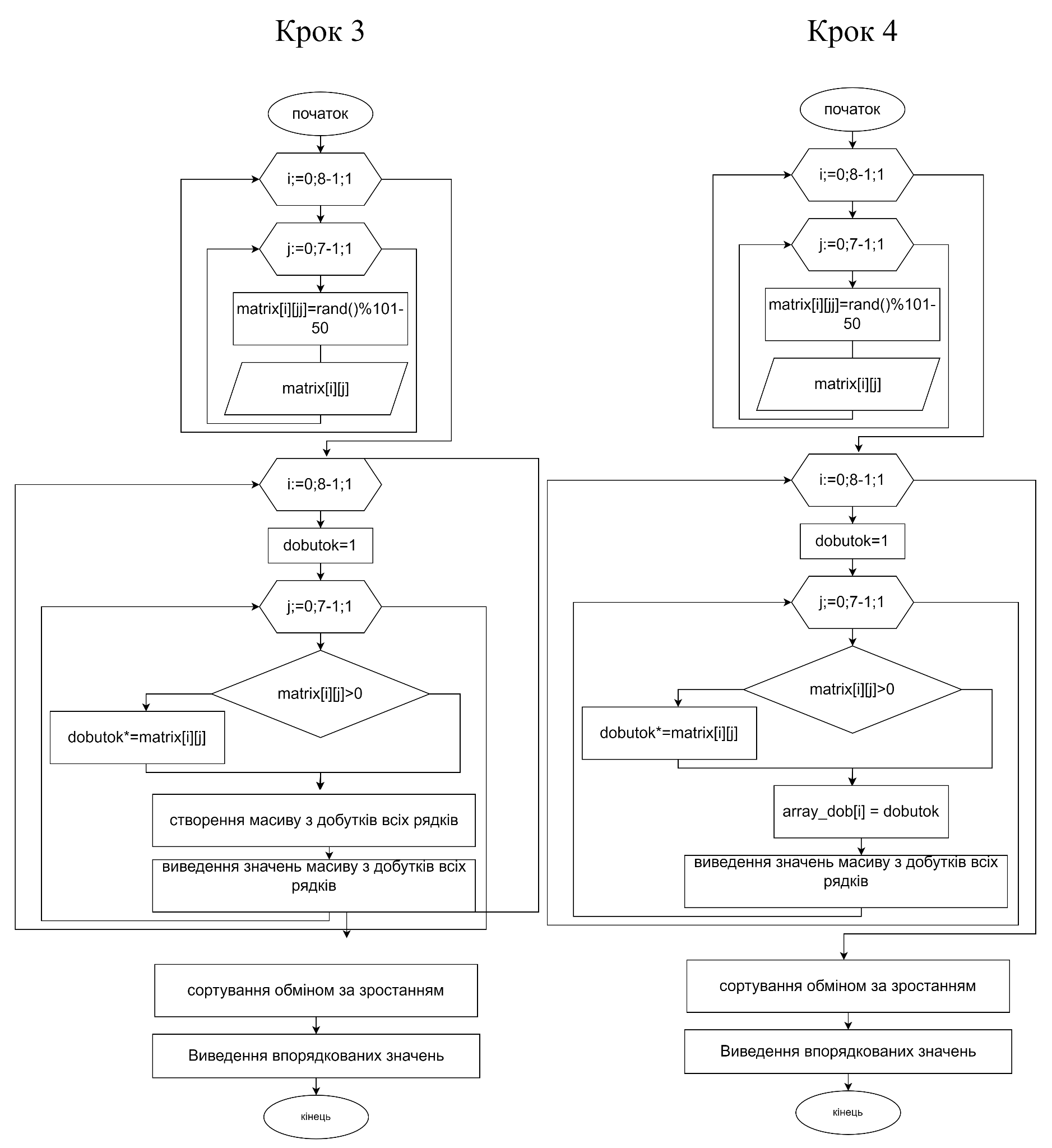
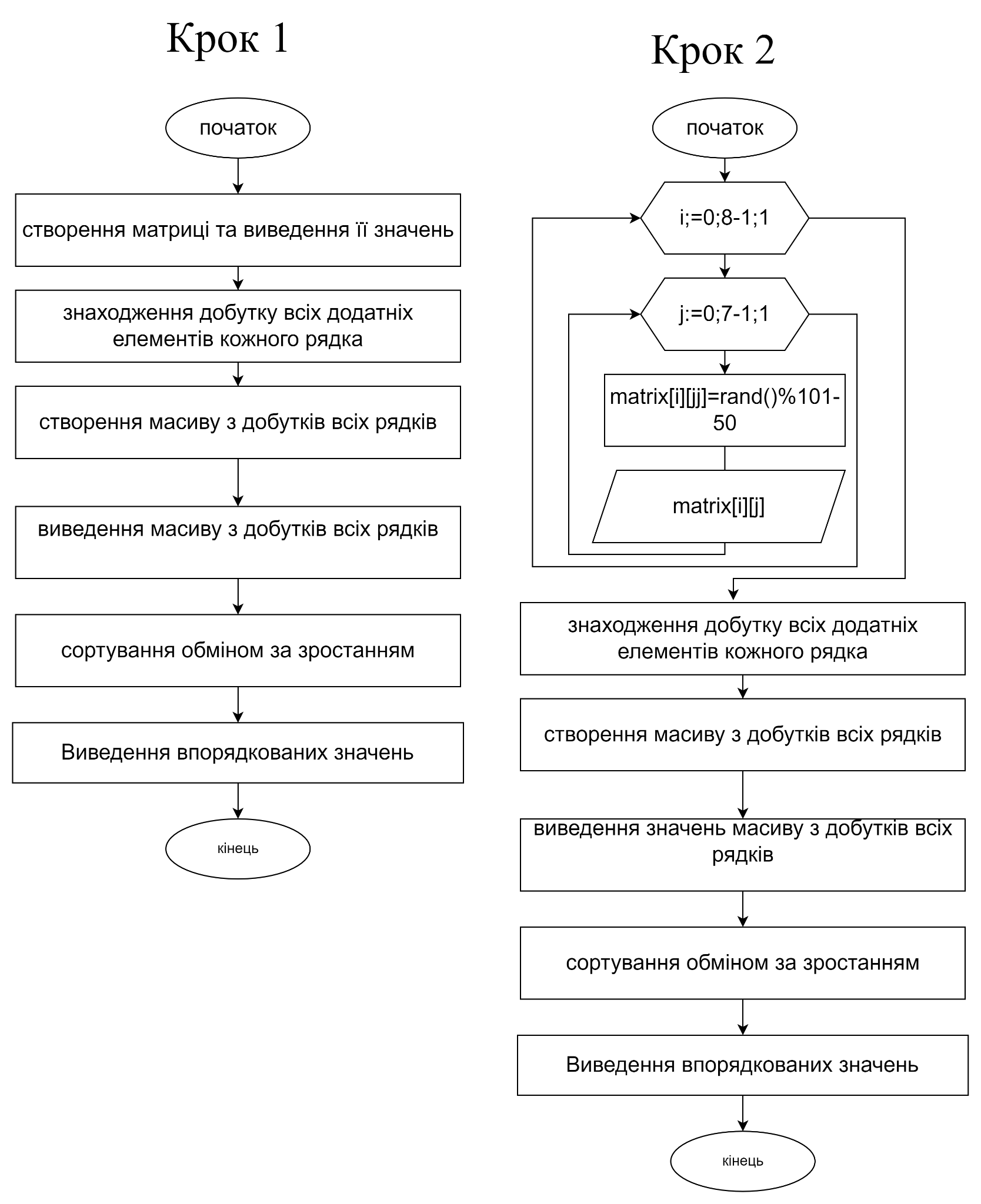
**Повторити для i** від i=0 до і=8-1 з кроком 1

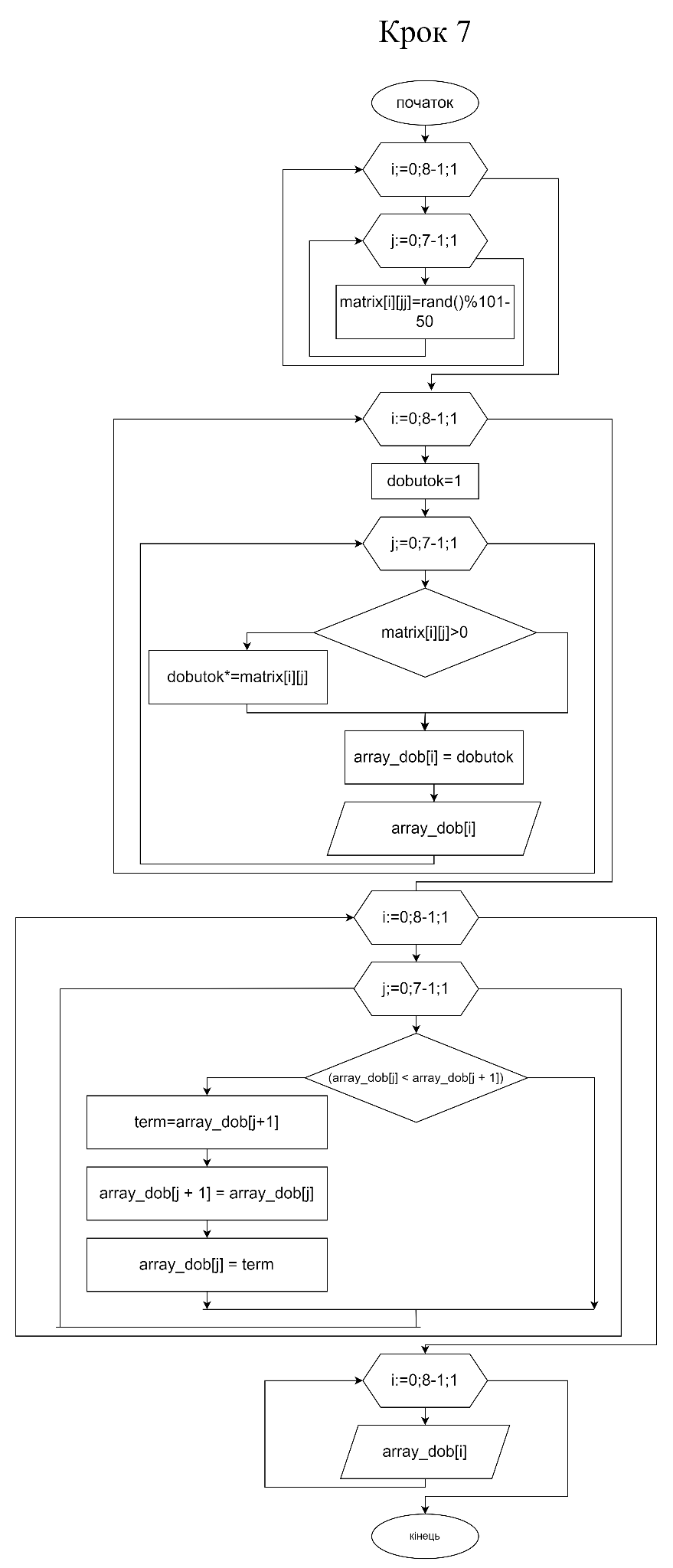
**Виведення** array\_dob[i]

**все повторити для і**

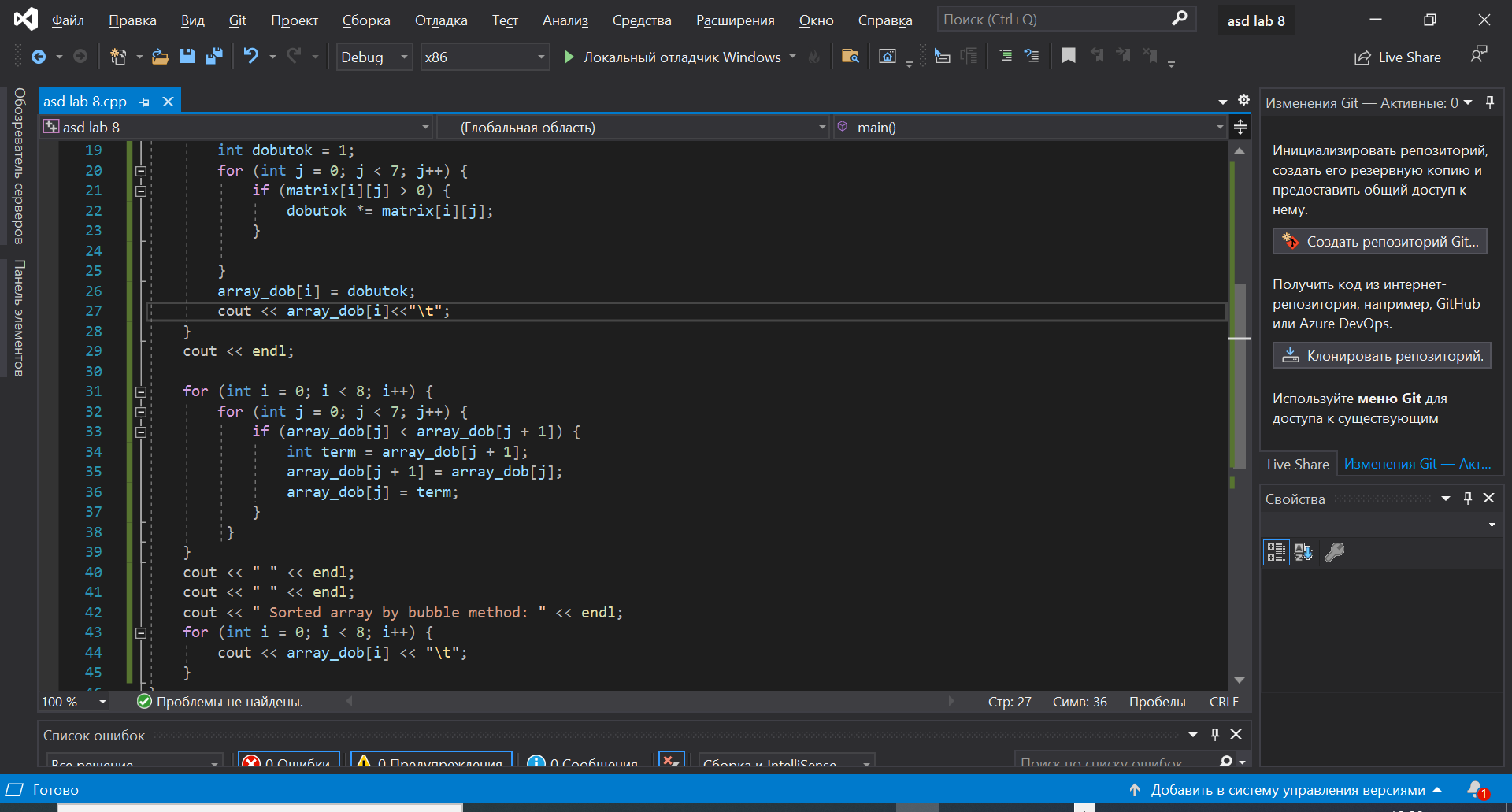
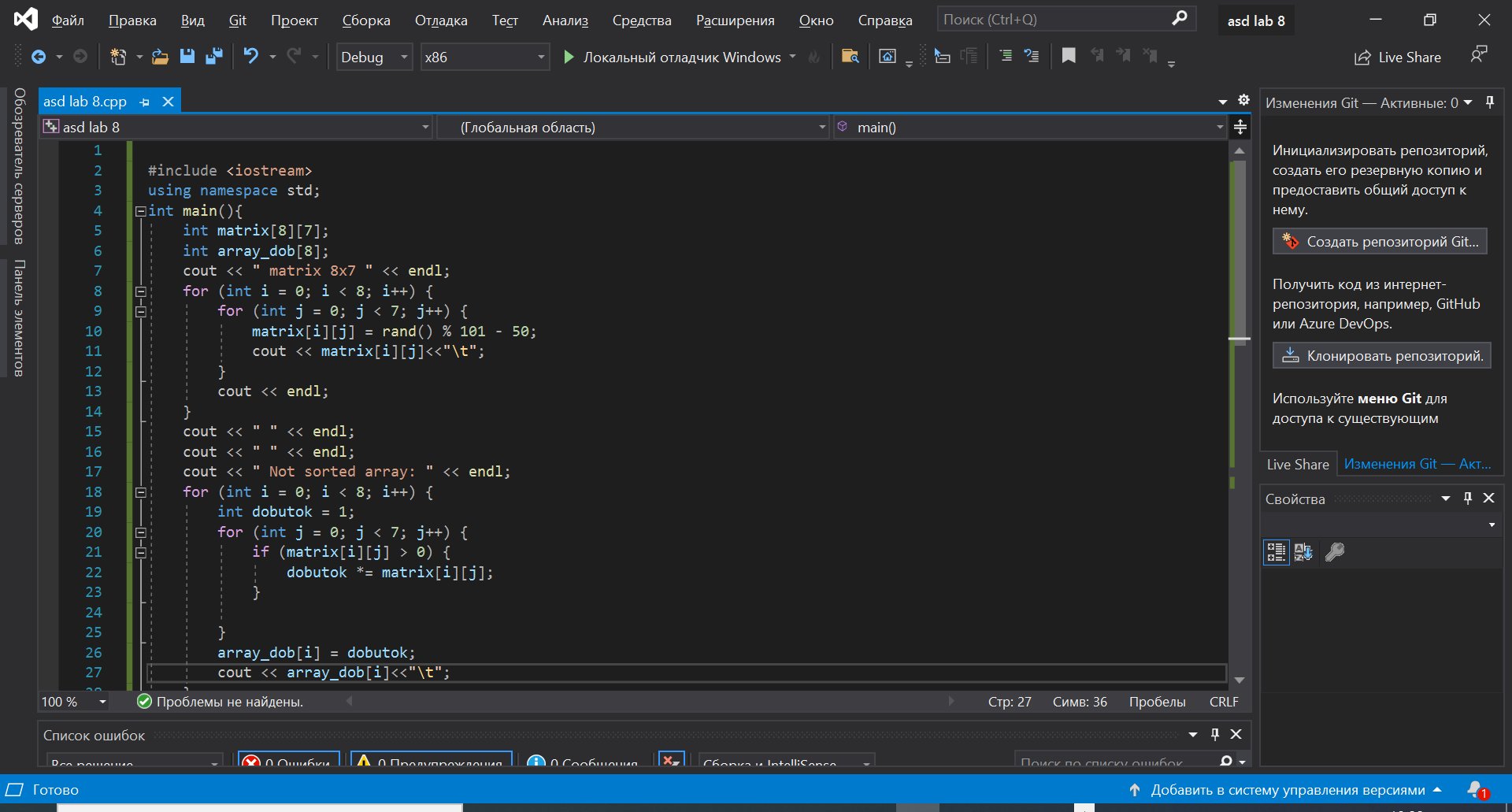
Кінець

1. Блок-схема алгоритму





1. Код програми



#include <iostream>

using namespace std;

int main(){

int matrix[8][7];

int array\_dob[8];

cout << " matrix 8x7 " << endl;

for (int i = 0; i < 8; i++) {

for (int j = 0; j < 7; j++) {

matrix[i][j] = rand() % 101 - 50;

cout << matrix[i][j]<<"\t";

}

cout << endl;

}

cout << " " << endl;

cout << " " << endl;

cout << " Not sorted array: " << endl;

for (int i = 0; i < 8; i++) {

int dobutok = 1;

for (int j = 0; j < 7; j++) {

if (matrix[i][j] > 0) {

dobutok \*= matrix[i][j];

}

}

array\_dob[i] = dobutok;

cout << array\_dob[i]<<"\t";

}

cout << endl;

for (int i = 0; i < 8; i++) {

for (int j = 0; j < 7; j++) {

if (array\_dob[j] < array\_dob[j + 1]) {

int term = array\_dob[j + 1];

array\_dob[j + 1] = array\_dob[j];

array\_dob[j] = term;

}

}

}

cout << " " << endl;

cout << " " << endl;

cout << " Sorted array by bubble method: " << endl;

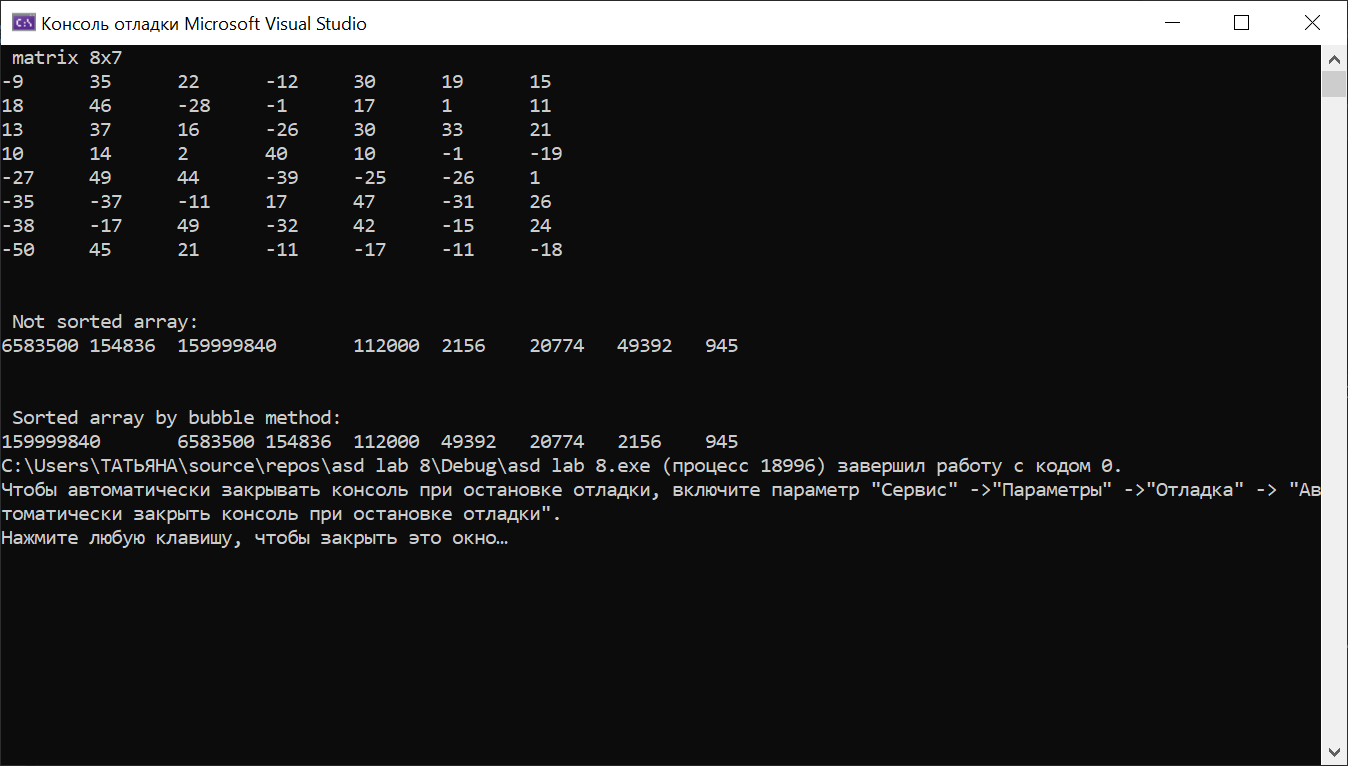
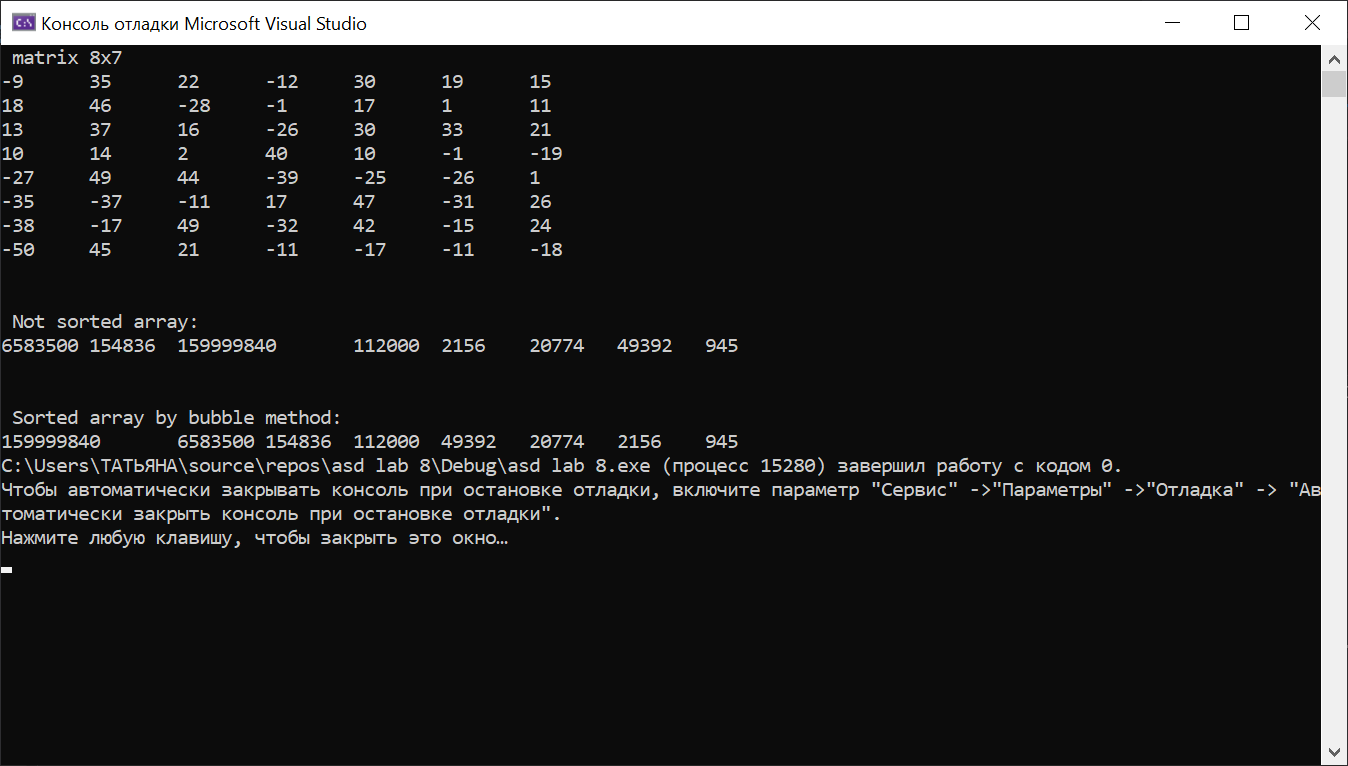
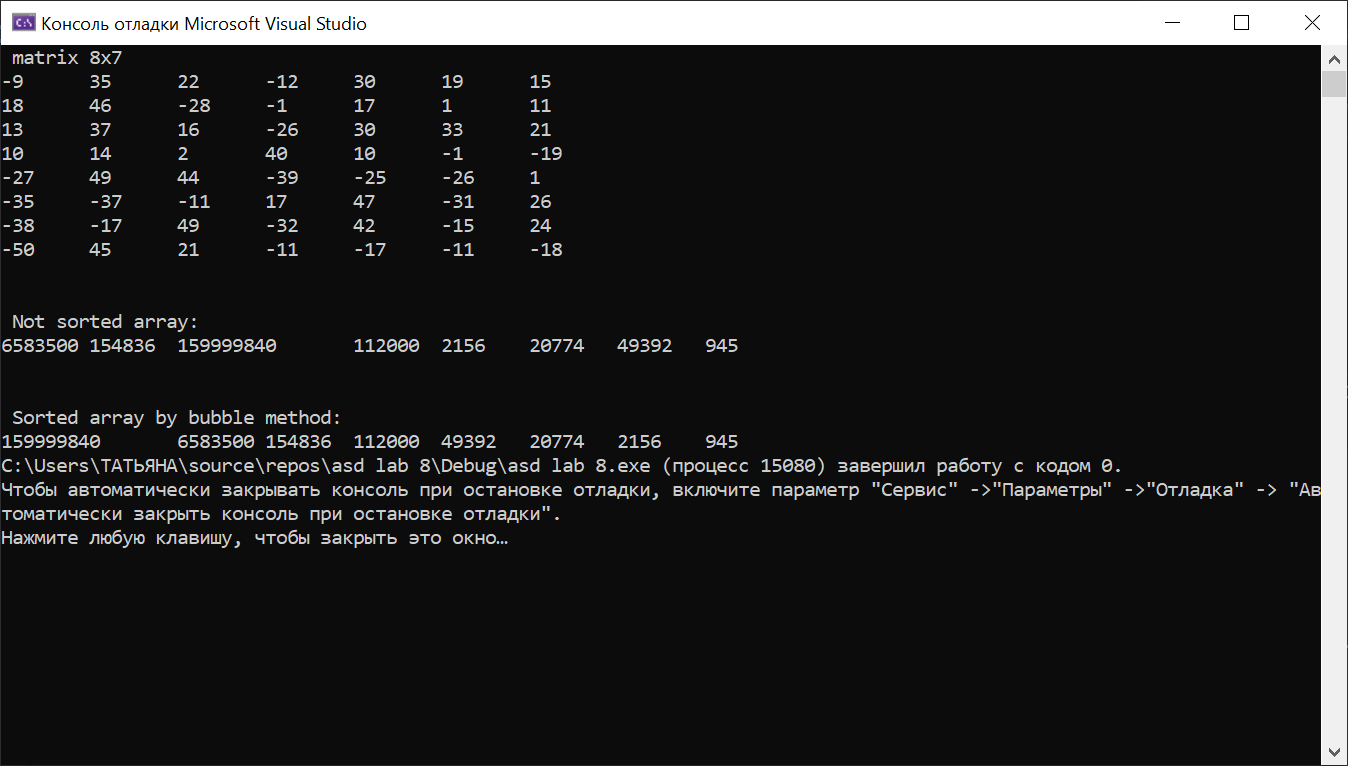
for (int i = 0; i < 8; i++) {

cout << array\_dob[i] << "\t";

}

}

1. Тестування програми



1. Висновки

Було досліджено алгоритми пошуку та сортування, набуто практичних навичок використання цих алгоритмів під час складання програмних специфікацій на прикладі завдання, у якому потребувалося створення двовимірного масиву та виконання операцій створення одновимірного з певними умовами. Під час даної лабораторної роботи було побудовано математичну модель, створений як псевдокод, так і код на мові с++, блок-схема алгоритму та його тестування.