Алгоритми та структури данних. Основи алгоритмів розгалуження.

*Додаток 1*

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 лінійного пошуку в послідовностях»

Варіант 15

Виконав студент ІП-12, Кириченко Владислав Сергійович

(шифр, прізвище, ім'я, по батькові)

Перевірив

( прізвище, ім'я, по батькові)

Київ 202 1

**Лабораторна робота №** 8

**Назва роботи**: Дослідження алгоритмів пошуку та сортування

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

**Варіант 15**

**Умова задачі***:*

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

1. Опису змінної індексованого типу (двовимірний масив) згідно з варіантом (*Розмірність -* ***8х5****. Тип данних елементів -* ***Дійсний***).
2. Ініціювання змінної, що описана в п.1 даного завдання.

3. Створення нової змінної індексованого типу (одновимірний масив) та її ініціювання значеннями, що обчислюються згідно з варіантом (*Із суми додатних значень елементів рядків двовимірного масиву. Відсортувати обміном* ***за зростанням***).

**Постановка задачі**:

*Початкові дані* **-** із початкових даних маємо лише розміз двовимірного масиву(***8х5***).

*Згенерувати двовимірний масив випадкових дійсних значень,виокремити з нього додатні елементи у окремий масив, та відсортувати цей масив за зростанням.*

*Результат* - одновимірний масив, відсортований **обміном за зростанням**.

**Побудова математичної моделі:**

Для реалізації алгоритму вирішення поставленої задачі нам портібен засіб генерації випадкового дійсного числа, нехай це буде функція ***randRealN()***, від

random real number(з англійської випадкове дійсне число). У коді програми буде використано стандартний метод генерації випадкового числа, саме: функція **rand().** Але т.я. ця функція повертає випадкове ціле значення, то дещо модивікуємо вираз і отримаємо **rand()%201 - 100 + double(rand()%100)/100**

(генерує дійсне число с проміжку [-100, 100] )

Складемо таблицю змінних:

|  |  |  |  |
| --- | --- | --- | --- |
| Змінна | Тип | Ім’я | Призначення |
| Двовимірний масив | індексований | ***arr2D*** | Проміжкове значення |
|  |  |  |  |
| Лічильник | цілочисельний | ***i*** | Проміжкове значення |
| Лічильник | цілочисельний | ***j*** | Проміжкове значення |
| формальний параметр(перший масив) | індексований | ***arr1*** | Проміжкове значення |
| формальний параметр(другий масив) | індексований | ***arr2*** | Проміжкове значення |
| формальний параметр для обміну значень між двома змінними | дійсний | ***a*** | Проміжкове значення |
| формальний параметр для обміну значень між двома змінними | дійсний | ***b*** | Проміжкове значення |
| одновимірний масив | індексований | ***arr1D*** | Результат |

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

**Крок 1.** Визначимо основні дії.

**Крок 2.**Деталізація ініціалізації ***arr2D , arr1D***

**Крок 3.**Деталізація заповнення масиву ***arr2D***

**Крок 4.**Деталізація заповнення масиву ***arr1D***

**Крок 5.**Деталізація сортування масиву ***arr1D***

*Псевдокод(основна прогрма):*

*Крок 1.*

**початок**

введення

ініціалізація ***arr2D , arr1D***

заповнення масиву ***arr2D***

заповнення масиву ***arr1D***

сортування масиву ***arr1D***

виведення

**кінець**

*Крок 2.*

**початок**

введення

***arr2D[8][5]***

***arr1D[5]***

заповнення масиву ***arr2D***

заповнення масиву ***arr1D***

сортування масиву ***arr1D***

виведення

**кінець**

*Крок 3.*

**початок**

введення

***arr2D[8][5]***

***arr1D[5]***

***fillArr2D(arr2D)***

заповнення масиву ***arr1D***

сортування масиву ***arr1D***

виведення

**кінець**

*Крок 4.*

**початок**

введення

***arr2D[8][5]***

***arr1D[5]***

***fillArr2D(arr2D)***

***fillArr1D(arr2D, arr1D)***

сортування масиву ***arr1D***

виведення

**кінець**

*Крок 5.*

**початок**

введення

***arr2D[8][5]***

***arr1D[5]***

***fillArr2D(arr2D)***

***fillArr1D(arr2D, arr1D)***

***bulbSort(arr1D)***

виведення

**кінець**

*Підпрограма* ***fillArr2D***

*Крок 1*

*функція* ***fillArr2D(arr2D[8][5])***

проходження по всіх елементах матриці за допомогою вкладеного масиву

присвоєння кожному елементу матриці випадкового дійсного значення

***кінець***

*Крок 2.*

*функція* ***fillArr2D(arr2D[8][5])***

***повторити***

***для і від 0 до 8 із кроком 1***

***повторити***

***для j від 0 до 5 із кроком 1***

присвоєння елементу матріці випадкового дійсного значення

***все повторити***

***все повторити***

***кінець***

*Крок 3.*

*функція* ***fillArr2D(arr2D[8][5])***

***повторити***

***для і від 0 до 8 із кроком 1***

***повторити***

***для j від 0 до 5 із кроком 1***

***arr2[i][j]= randRealN()***

***все повторити***

***все повторити***

***кінець***

*Підпрограма* ***fillArr1D***

*Крок 1*

*функція* ***fillArr1D(arr2D[8][5], arr1D[8])***

проходження по всіх рядках матриці та елементах одновимірного масива за допомогою зовншнього циклу

присвоєння значення 0 поточному елементу масиву

проходження по всіх елементах рядка матриці

перевірка чи поточний елемент матриці більше за 0

збільшення поточного елементу масива

***кінець***

*Крок 2.*

*функція* ***fillArr1D(arr2D[8][5], arr1D[8])***

***повторити***

***для і від 0 до 8 із кроком 1***

присвоєння значення 0 поточному елементу масиву

проходження по всіх елементах рядка матриці

перевірка чи поточний елемент матриці більше за 0

збільшення поточного елементу масива

***все повторити***

***кінець***

*Крок 3.*

*функція* ***fillArr1D(arr2D[8][5], arr1D[8])***

***повторити***

***для і від 0 до 8 із кроком 1***

***arr1[i] = 0***

проходження по всіх елементах рядка матриці

перевірка чи поточний елемент матриці більше за 0

збільшення поточного елементу масива

***все повторити***

***кінець***

*Крок 4.*

*функція* ***fillArr1D(arr2D[8][5], arr1D[8])***

***повторити***

***для і від 0 до 8 із кроком 1***

***arr1[i] = 0***

***повторити***

***для j від 0 до 5 із кроком 1***

перевірка чи поточний елемент матриці більше за 0

збільшення поточного елементу масива

***все повторити***

***все повторити***

***кінець***

*Крок 5.*

*функція* ***fillArr1D(arr2D[8][5], arr1D[8])***

***повторити***

***для і від 0 до 8 із кроком 1***

***arr1[i] = 0***

***повторити***

***для j від 0 до 5 із кроком 1***

***якщо arr2[i][j] > 0***

***то***

збільшення поточного елементу масива

***все повторити***

***все повторити***

***кінець***

*Крок 6.*

*функція* ***fillArr1D(arr2D[8][5], arr1D[8])***

***повторити***

***для і від 0 до 8 із кроком 1***

***arr1[i] = 0***

***повторити***

***для j від 0 до 5 із кроком 1***

***якщо arr2[i][j] > 0***

***то***

***arr1[i]+=arr2[i][j]***

***все повторити***

***все повторити***

***кінець***

*Підпрограма* ***bulbSort***

*Крок 1*

*функція* ***bulbSort(arr1D[8])***

цикл що повторється 8 разів(кільк елементів масиву)

проходження по всім парам елементів

перевірка чи пара впорядкована

перестановка елементів

***Кінець***

*Крок 2*

*функція* ***bulbSort(arr1D[8])***

***повторити***

***для і від 0 до 8 із кроком 1***

проходження по всім парам елементів

перевірка чи пара впорядкована

перестановка елементів

***все повторити***

***кінець***

*Крок 3*

*функція* ***bulbSort(arr1D[8])***

***повторити***

***для і від 0 до 8 із кроком 1***

***повторити***

***для j від 0 до 7 із кроком 1***

перевірка чи пара впорядкована

перестановка елементів

***все повторити***

***все повторити***

***кінець***

*Крок 4*

*функція* ***bulbSort(arr1D[8])***

***повторити***

***для і від 0 до 8 із кроком 1***

***повторити***

***для j від 0 до 7 із кроком 1***

***якщо arr1[j] > arr1[j+1]***

***то***

перестановка елементів

***все якщо***

***все повторити***

***все повторити***

***кінець***

*Крок 5*

*функція* ***bulbSort(arr1D[8])***

***повторити***

***для і від 0 до 8 із кроком 1***

***повторити***

***для j від 0 до 7 із кроком 1***

***якщо arr1[j] > arr1[j+1]***

***то***

***swap(&arr1[j], &arr1[j+1])***

***все якщо***

***все повторити***

***все повторити***

***кінець***

*Підпрограма* ***swap***

*Крок 1*

*функція* ***swap(a,b)***

***кінець***

*Крок 2*

*функція* ***swap(a,b)***

**a = a + b**

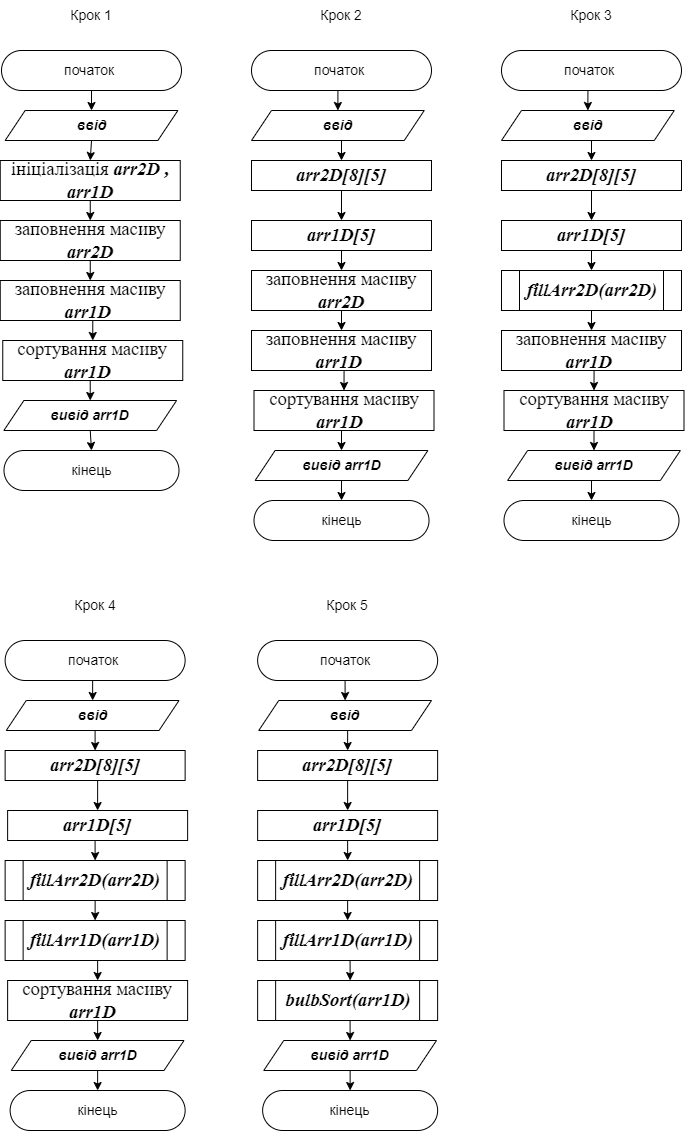
**b = a - b**

**a = a - b**

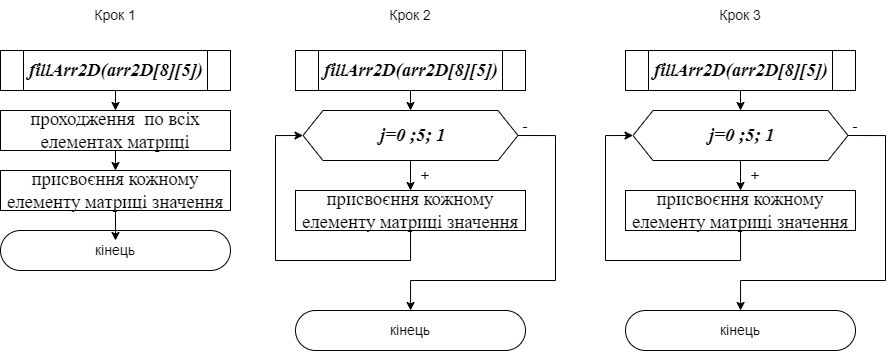
**кінець**

Блок схема:

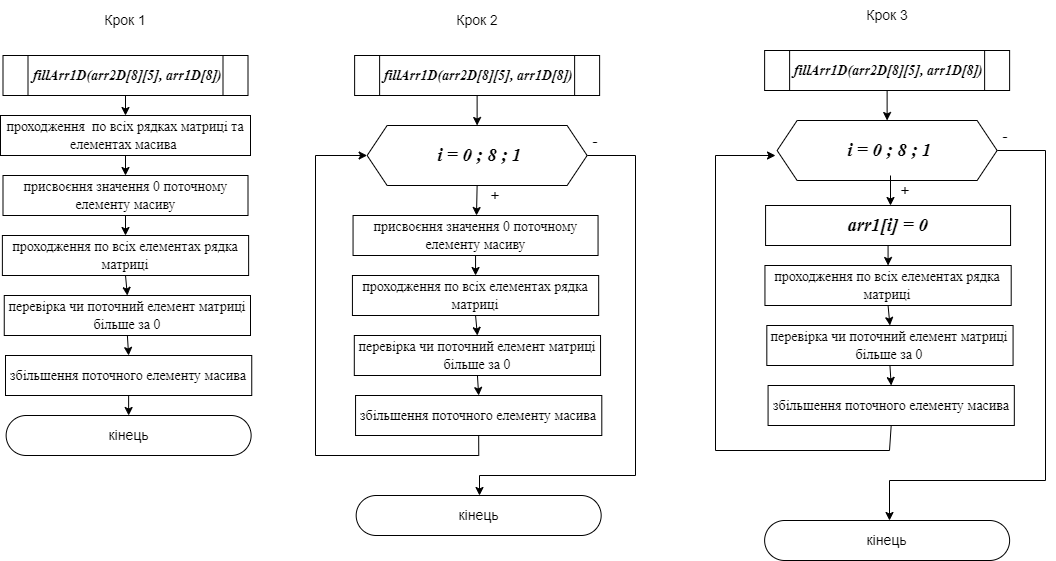
*Основна програма*

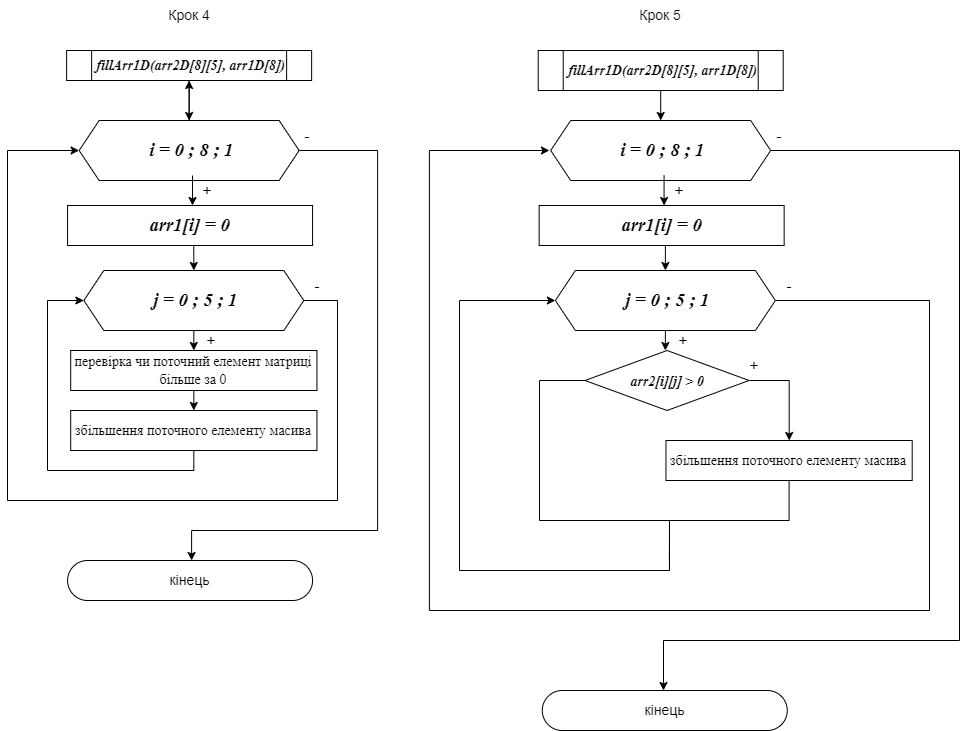


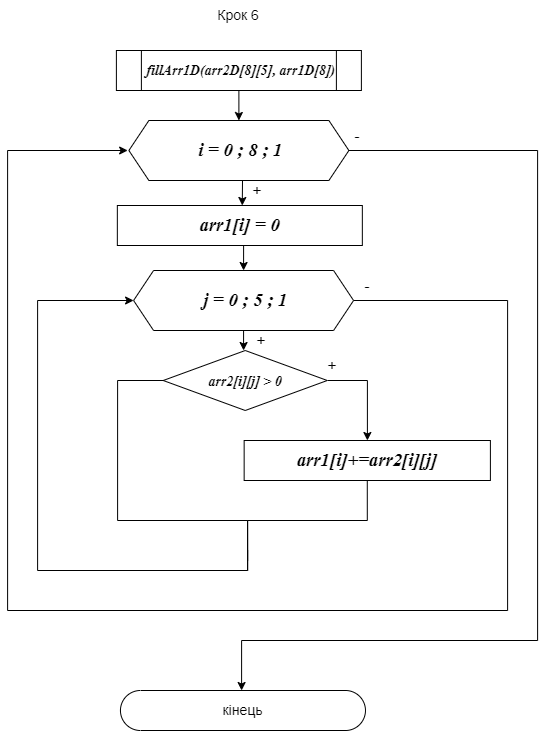
*Підпрограма* ***fillArr2D***



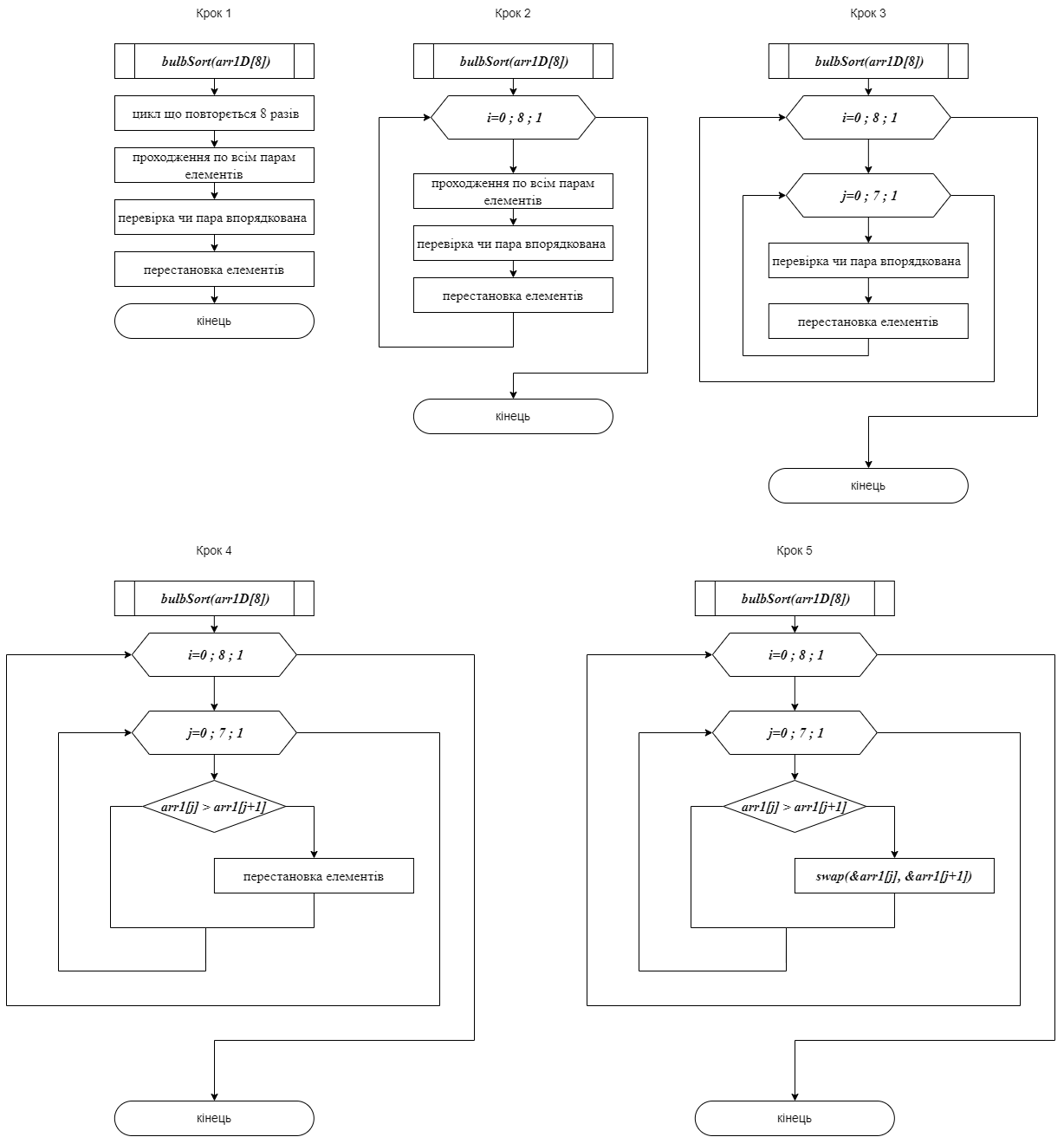
*Підпрограма* ***fillArr1D***



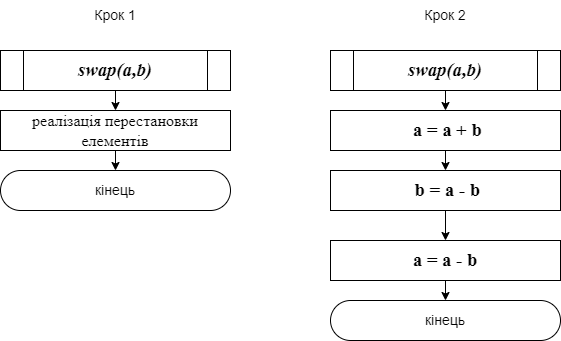




*Підпрограма* ***bulbSort***



*Підпрограма* ***swap***



1. Код програми(С++)

#include <iostream>

#include <iomanip>

using namespace std;

// Заповнення масивів

void fillArr1D(double [8][5],double [8]);

void fillArr2D(double [8][5]);

// Відображення масивів

void displayArr2D(double [8][5], string);

void displayArr1D(double [8],string);

// Сортування масивів

void swap(double\* , double\* );

void bulbSort(double [8]);

int main() {

// Ініціалізація масивів

double arr2D[8][5];

double arr1D[8];

// Заповнення масивів

fillArr2D(arr2D);

fillArr1D(arr2D,arr1D);

// Відбраження початкових масивів

displayArr2D(arr2D, "Двовимірний масив: ");

displayArr1D(arr1D, "Початковий одновимірний масив: ");

//Сортування одновимірного масиву

bulbSort(arr1D);

//Відображення відосртованого масиву

displayArr1D(arr1D, "Відсортований одновимірний масив: ");

}

void fillArr1D(double arr2[8][5],double arr1[8] ) {

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

arr1[i] = 0;

for (int j = 0 ; j < 5; j++) {

if (arr2[i][j] > 0 ){

arr1[i]+=arr2[i][j];

}

}

}

}

void fillArr2D(double arr2[8][5]) {

srand(time(NULL));

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

for (int j = 0; j < 5; j++) {

arr2[i][j]=rand()%201 -100 + double(rand()%100)/100;

}

}

}

void displayArr1D(double arr1[8], string message){

cout << message << "\n\n";

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

cout << setw(3) << arr1[i] << "\n";;

}

cout << "\n";

}

void displayArr2D(double arr2[8][5], string message){

cout << message << "\n\n";

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

for (int j = 0; j < 5; j++) {

cout <<setw(6)<< arr2[i][j] << " ";

}

cout << "\n";;

}

cout << "\n";

}

void bulbSort(double arr1[8]) {

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

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

if (arr1[j] > arr1[j+1]){

swap(&arr1[j], &arr1[j+1]);

}

}

}

}

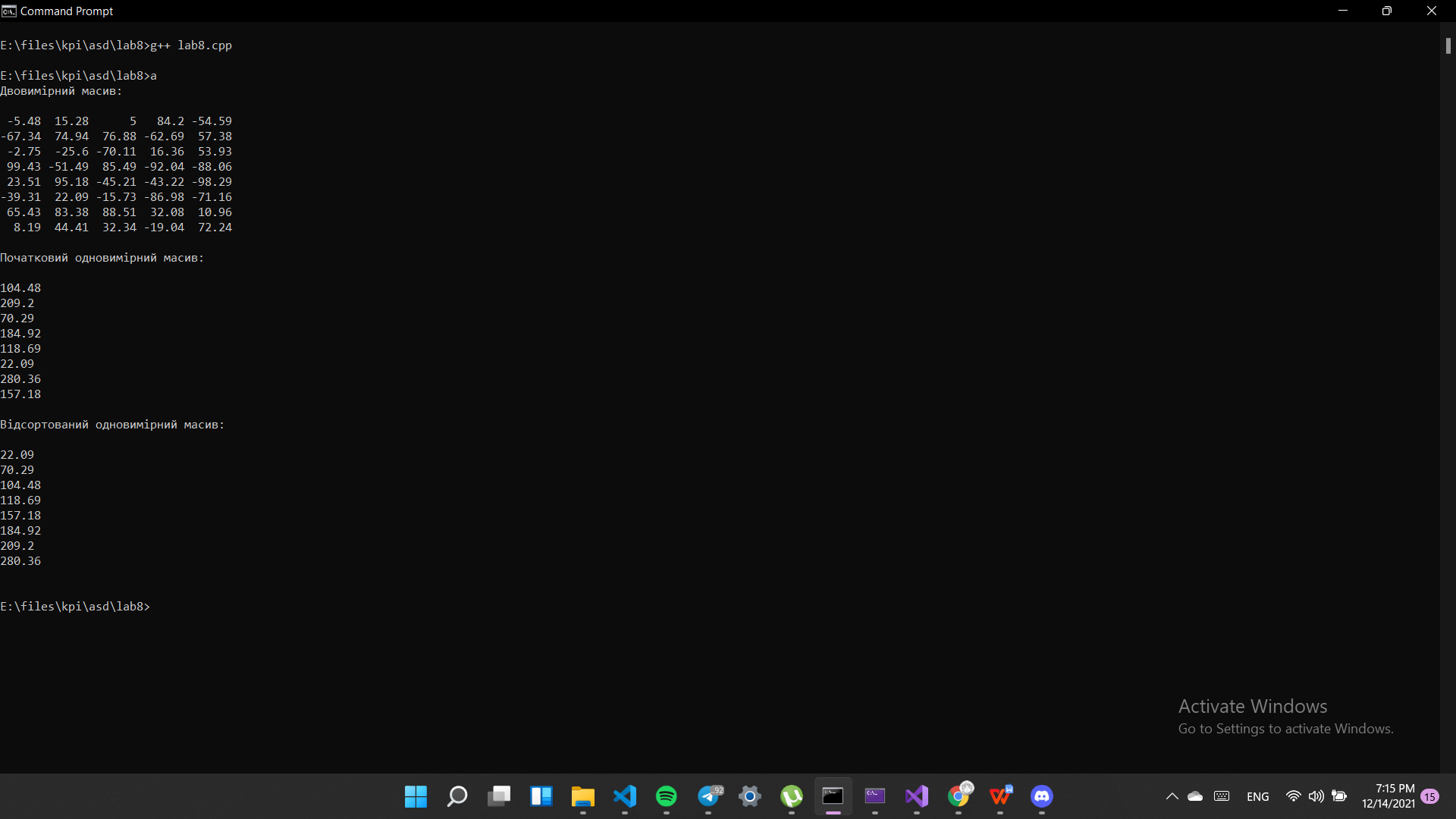
void swap(double\* a, double\* b) {

\*a += \*b;

\*b = \*a - \*b;

\*a = \*a - \*b;

}



ПЕРЕВІРКА

arr2D[8][5]

arr1D[8]

fillArr2D(arr2D)

перебіг дій у піпрограмі цикл

ітерація зовн. цикла: 0

ітерація внутр. цикла: 0

arr2[0][0] = 76.34

ітерація внутр. цикла: 1

arr2[0][1] = 51.69

ітерація внутр. цикла: 2

arr2[0][2] = 92.68

ітерація внутр. цикла: 3

arr2[0][3] = 56.76

ітерація внутр. цикла: 4

arr2[0][4] = -33.39

все цикл

ітерація зовн. цикла: 1

ітерація внутр. цикла: 0

arr2[1][0] = 78.88

ітерація внутр. цикла: 1

arr2[1][1] = 29.19

ітерація внутр. цикла: 2

arr2[1][2] = -65.42

ітерація внутр. цикла: 3

arr2[1][3] = -74.57

ітерація внутр. цикла: 4

arr2[1][4] = -3.82

все цикл

ітерація зовн. цикла: 2

ітерація внутр. цикла: 0

arr2[2][0] = -68.43

ітерація внутр. цикла: 1

arr2[2][1] = 33.08

ітерація внутр. цикла: 2

arr2[2][2] = -8.36

ітерація внутр. цикла: 3

arr2[2][3] = -93

ітерація внутр. цикла: 4

arr2[2][4] = 59.68

все цикл

ітерація зовн. цикла: 3

ітерація внутр. цикла: 0

arr2[3][0] = 25.31

ітерація внутр. цикла: 1

arr2[3][1] = 5.39

ітерація внутр. цикла: 2

arr2[3][2] = 78.82

ітерація внутр. цикла: 3

arr2[3][3] = 40.48

ітерація внутр. цикла: 4

arr2[3][4] = -12.26

все цикл

ітерація зовн. цикла: 4

ітерація внутр. цикла: 0

arr2[4][0] = -68.07

ітерація внутр. цикла: 1

arr2[4][1] = -38.71

ітерація внутр. цикла: 2

arr2[4][2] = -55.5

ітерація внутр. цикла: 3

arr2[4][3] = 39.05

ітерація внутр. цикла: 4

arr2[4][4] = -80.3

все цикл

ітерація зовн. цикла: 5

ітерація внутр. цикла: 0

arr2[5][0] = 57.33

ітерація внутр. цикла: 1

arr2[5][1] = -80.06

ітерація внутр. цикла: 2

arr2[5][2] = -96.24

ітерація внутр. цикла: 3

arr2[5][3] = 43.66

ітерація внутр. цикла: 4

arr2[5][4] = 50.26

все цикл

ітерація зовн. цикла: 6

ітерація внутр. цикла: 0

arr2[6][0] = -95.39

ітерація внутр. цикла: 1

arr2[6][1] = -5.82

ітерація внутр. цикла: 2

arr2[6][2] = 67.69

ітерація внутр. цикла: 3

arr2[6][3] = -19.8

ітерація внутр. цикла: 4

arr2[6][4] = 8.14

все цикл

ітерація зовн. цикла: 7

ітерація внутр. цикла: 0

arr2[7][0] = -10.6

ітерація внутр. цикла: 1

arr2[7][1] = 87.82

ітерація внутр. цикла: 2

arr2[7][2] = -77.15

ітерація внутр. цикла: 3

arr2[7][3] = -88.61

ітерація внутр. цикла: 4

arr2[7][4] = 81.29

все цикл

цикл

ітерація зовн. цикла: 0

arr1[0] = 0

ітерація внутр. цикла: 0

arr2[i][j] > 0 - 1

arr1[0]+=76.34ітерація внутр. цикла: 1

arr2[i][j] > 0 - 1

arr1[0]+=51.69ітерація внутр. цикла: 2

arr2[i][j] > 0 - 1

arr1[0]+=92.68ітерація внутр. цикла: 3

arr2[i][j] > 0 - 1

arr1[0]+=56.76ітерація внутр. цикла: 4

arr2[i][j] > 0 - 0

ітерація зовн. цикла: 1

arr1[1] = 0

ітерація внутр. цикла: 0

arr2[i][j] > 0 - 1

arr1[1]+=78.88ітерація внутр. цикла: 1

arr2[i][j] > 0 - 1

arr1[1]+=29.19ітерація внутр. цикла: 2

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 3

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 4

arr2[i][j] > 0 - 0

ітерація зовн. цикла: 2

arr1[2] = 0

ітерація внутр. цикла: 0

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 1

arr2[i][j] > 0 - 1

arr1[2]+=33.08ітерація внутр. цикла: 2

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 3

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 4

arr2[i][j] > 0 - 1

arr1[2]+=59.68ітерація зовн. цикла: 3

arr1[3] = 0

ітерація внутр. цикла: 0

arr2[i][j] > 0 - 1

arr1[3]+=25.31ітерація внутр. цикла: 1

arr2[i][j] > 0 - 1

arr1[3]+=5.39ітерація внутр. цикла: 2

arr2[i][j] > 0 - 1

arr1[3]+=78.82ітерація внутр. цикла: 3

arr2[i][j] > 0 - 1

arr1[3]+=40.48ітерація внутр. цикла: 4

arr2[i][j] > 0 - 0

ітерація зовн. цикла: 4

arr1[4] = 0

ітерація внутр. цикла: 0

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 1

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 2

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 3

arr2[i][j] > 0 - 1

arr1[4]+=39.05ітерація внутр. цикла: 4

arr2[i][j] > 0 - 0

ітерація зовн. цикла: 5

arr1[5] = 0

ітерація внутр. цикла: 0

arr2[i][j] > 0 - 1

arr1[5]+=57.33ітерація внутр. цикла: 1

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 2

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 3

arr2[i][j] > 0 - 1

arr1[5]+=43.66ітерація внутр. цикла: 4

arr2[i][j] > 0 - 1

arr1[5]+=50.26ітерація зовн. цикла: 6

arr1[6] = 0

ітерація внутр. цикла: 0

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 1

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 2

arr2[i][j] > 0 - 1

arr1[6]+=67.69ітерація внутр. цикла: 3

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 4

arr2[i][j] > 0 - 1

arr1[6]+=8.14ітерація зовн. цикла: 7

arr1[7] = 0

ітерація внутр. цикла: 0

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 1

arr2[i][j] > 0 - 1

arr1[7]+=87.82ітерація внутр. цикла: 2

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 3

arr2[i][j] > 0 - 0

ітерація внутр. цикла: 4

arr2[i][j] > 0 - 1

arr1[7]+=81.29bulbSort(arr1D)

перебіг дій у піпрограмі цикл

ітерація зовн. цикла: 0

ітерація внутр. цикла: 0

arr1[j] > arr1[j+1] - 1

swap(arr1[0], arr1[1])

перебіг підпрограми swap

a+=108.07

b = 169.4

a =169.4

ітерація внутр. цикла: 1

arr1[j] > arr1[j+1] - 1

swap(arr1[1], arr1[2])

перебіг підпрограми swap

a+=92.76

b = 184.71

a =184.71

ітерація внутр. цикла: 2

arr1[j] > arr1[j+1] - 1

swap(arr1[2], arr1[3])

перебіг підпрограми swap

a+=150

b = 127.47

a =127.47

ітерація внутр. цикла: 3

arr1[j] > arr1[j+1] - 1

swap(arr1[3], arr1[4])

перебіг підпрограми swap

a+=39.05

b = 238.42

a =238.42

ітерація внутр. цикла: 4

arr1[j] > arr1[j+1] - 1

swap(arr1[4], arr1[5])

перебіг підпрограми swap

a+=151.25

b = 126.22

a =126.22

ітерація внутр. цикла: 5

arr1[j] > arr1[j+1] - 1

swap(arr1[5], arr1[6])

перебіг підпрограми swap

a+=75.83

b = 201.64

a =201.64

ітерація внутр. цикла: 6

arr1[j] > arr1[j+1] - 1

swap(arr1[6], arr1[7])

перебіг підпрограми swap

a+=169.11

b = 108.36

a =108.36

ітерація зовн. цикла: 1

ітерація внутр. цикла: 0

arr1[j] > arr1[j+1] - 1

swap(arr1[0], arr1[1])

перебіг підпрограми swap

a+=92.76

b = 15.31

a =15.31

ітерація внутр. цикла: 1

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 2

arr1[j] > arr1[j+1] - 1

swap(arr1[2], arr1[3])

перебіг підпрограми swap

a+=39.05

b = 110.95

a =110.95

ітерація внутр. цикла: 3

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 4

arr1[j] > arr1[j+1] - 1

swap(arr1[4], arr1[5])

перебіг підпрограми swap

a+=75.83

b = 75.42

a =75.42

ітерація внутр. цикла: 5

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 6

arr1[j] > arr1[j+1] - 0

ітерація зовн. цикла: 2

ітерація внутр. цикла: 0

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 1

arr1[j] > arr1[j+1] - 1

swap(arr1[1], arr1[2])

перебіг підпрограми swap

a+=39.05

b = 69.02

a =69.02

ітерація внутр. цикла: 2

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 3

arr1[j] > arr1[j+1] - 1

swap(arr1[3], arr1[4])

перебіг підпрограми swap

a+=75.83

b = 74.17

a =74.17

ітерація внутр. цикла: 4

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 5

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 6

arr1[j] > arr1[j+1] - 0

ітерація зовн. цикла: 3

ітерація внутр. цикла: 0

arr1[j] > arr1[j+1] - 1

swap(arr1[0], arr1[1])

перебіг підпрограми swap

a+=39.05

b = 53.71

a =53.71

ітерація внутр. цикла: 1

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 2

arr1[j] > arr1[j+1] - 1

swap(arr1[2], arr1[3])

перебіг підпрограми swap

a+=75.83

b = 32.24

a =32.24

ітерація внутр. цикла: 3

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 4

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 5

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 6

arr1[j] > arr1[j+1] - 0

ітерація зовн. цикла: 4

ітерація внутр. цикла: 0

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 1

arr1[j] > arr1[j+1] - 1

swap(arr1[1], arr1[2])

перебіг підпрограми swap

a+=75.83

b = 16.93

a =16.93

ітерація внутр. цикла: 2

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 3

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 4

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 5

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 6

arr1[j] > arr1[j+1] - 0

ітерація зовн. цикла: 5

ітерація внутр. цикла: 0

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 1

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 2

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 3

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 4

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 5

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 6

arr1[j] > arr1[j+1] - 0

ітерація зовн. цикла: 6

ітерація внутр. цикла: 0

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 1

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 2

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 3

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 4

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 5

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 6

arr1[j] > arr1[j+1] - 0

ітерація зовн. цикла: 7

ітерація внутр. цикла: 0

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 1

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 2

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 3

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 4

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 5

arr1[j] > arr1[j+1] - 0

ітерація внутр. цикла: 6

arr1[j] > arr1[j+1] - 0

все цикл

**Висновок -** Було досліджено алгоритми пошуку та сортування, набуто практичних навичок використання цих алгоритмів під час складання програмних специфікацій.

Декомпозовано задачу на 4 етапи:

1. Генерація матриці.

2. Генерація масиву, з суми додатніх елементів кожого рядка матриці

3. Сортування отриманого масиву, за допомогую сортування обміном

4.Обмін значень двох змінних (потрібно для сортування обміном)