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 34

Виконав студент ІП-15, Чінь Хоанг Вьет
Перевірив Вечерковська Анастасія Сергіївна

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.

Варіант 34

Постановка задачі

Завдання

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

1. Опису змінної індексованого типу (двовимірний масив) згідно з варіантом (табл. 1).
2. Ініціювання змінної, що описана в п.1 даного завдання.
3. Обчислення змінної, що описана в п.1, згідно з варіантом (табл. 1).

№	Опис варіанту
34	Задано матрицю дійсних чисел A[m,n]. При обході матриці по рядках знайти в ній останній мінімальний елемент X і його місцезнаходження. Порівняти значення X із середньоарифметичним значенням елементів під побічною діагоналлю.

Розв'язання: Спочатку, створимо двовимірний масив з **n** рядків та стовпчиків. Потім, знайдемо середньоарифметичну суму елементів під побічною діагоналлю за допомогою підпрограми. Після цього створимо ще одну підпрограму, яка знаходить місце розташування та мінімальний елемент у матриці. І нарешті, порівняємо мінімальне значення матриці та середньоарифметичне під побічною діагоналлю

Математична модель:

Змінна	Тип	Ім'я	Призначення
Рядки та стовпці	Дійсний	rowcol	Початкові дані
Двовимірний масив	Дійсний	Arr	Початкові дані
Початкове значення масиву	Дійсний	S	Початкові дані
Обхід масиву	Цілий та ненульовий	backward	Початкові дані
Середньоарифметична сума	Дійсний	Sum	Початкові дані, проміжні дані
Кількість елементів	Цілий	count	Початкові дані,

під поб. діагоналлю			проміжні дані
Лічильник 1	Цілочисельний та натуральний	i	Параметр циклу
Лічильник 2	Цілочисельний та натуральний	j	Параметр циклу
Значення для функцій	Цілочисельний та натуральний	n	Початкові дані
Розташування мін. значення по рядку	Цілочисельний та натуральний	min_row	Початкові дані, проміжні дані
Розташування мін. значення по стовпцю	Цілочисельний та натуральний	min_col	Початкові дані, проміжні дані
Створення матриці	Процедура	input	Початкові дані
Знаходження суми	Процедура	sum	Початкові дані
Знаходження мін. ел.	Процедура	search_min	Початкові дані
Порівняння значень	Процедура	comparison	Початкові дані

1. Визначаємо основні дії
2. Вводимо значення rowcol та ініціалізація початкових даних
3. Створення підпрограми input для створення масиву
4. Створення підпрограми sum для знаходження суми елементів
5. Створення підпрограми search_min знаходження мін. елемента
6. Створення підпрограми comparison для порівняння значень

Псевдокод:

Головна програма:

Ввід rowcol;

Arr = input(rowcol);

Sum = sum(Arr, rowcol);

min_value = search_min(Arr, rowcol);

comparison(sum, min value);

Підпрограми:

Початок input

input(n)

S = 1;

backward = 1;

для **i** від **0** до **n повторити**

Якщо backward < 0

для **j** від **n-1** до **0 повторити**

Arr[i][j] = S++;

Все повторити

Все якщо

Інакше

для **j** від **0** до **n повторити**

Arr[i][j] = S++;

Все повторити

Все інакше

backward = -backward

Все повторити

return Arr

Кінець input

Початок sum

sum(arr, n)

sum = 0;

count = 0;

для **i** від **0** до **n повторити**

для **j** від **0** до **n повторити**

Якщо j + I > n-1

sum += arr[i][j];

count++

Все якщо

Все повторити

Все повторити

sum /= count;

return sum;

Кінець sum

Початок min_value

min_value(arr, n)

min = 1000;

min_row = 0;

min_col = 0;

для **i** від **0** до **n повторити**

для **j** від **0** до **n повторити**

Якщо arr[i][j] < min

Min = arr[i][j];

min_row = j;

min_col = i;

Все якщо

Все повторити

Все повторити

Return min

Кінець min_value

Початок comparison

comparison(sum, min)

Якщо sum == min;

Вивід “Числа рівні”

Все якщо

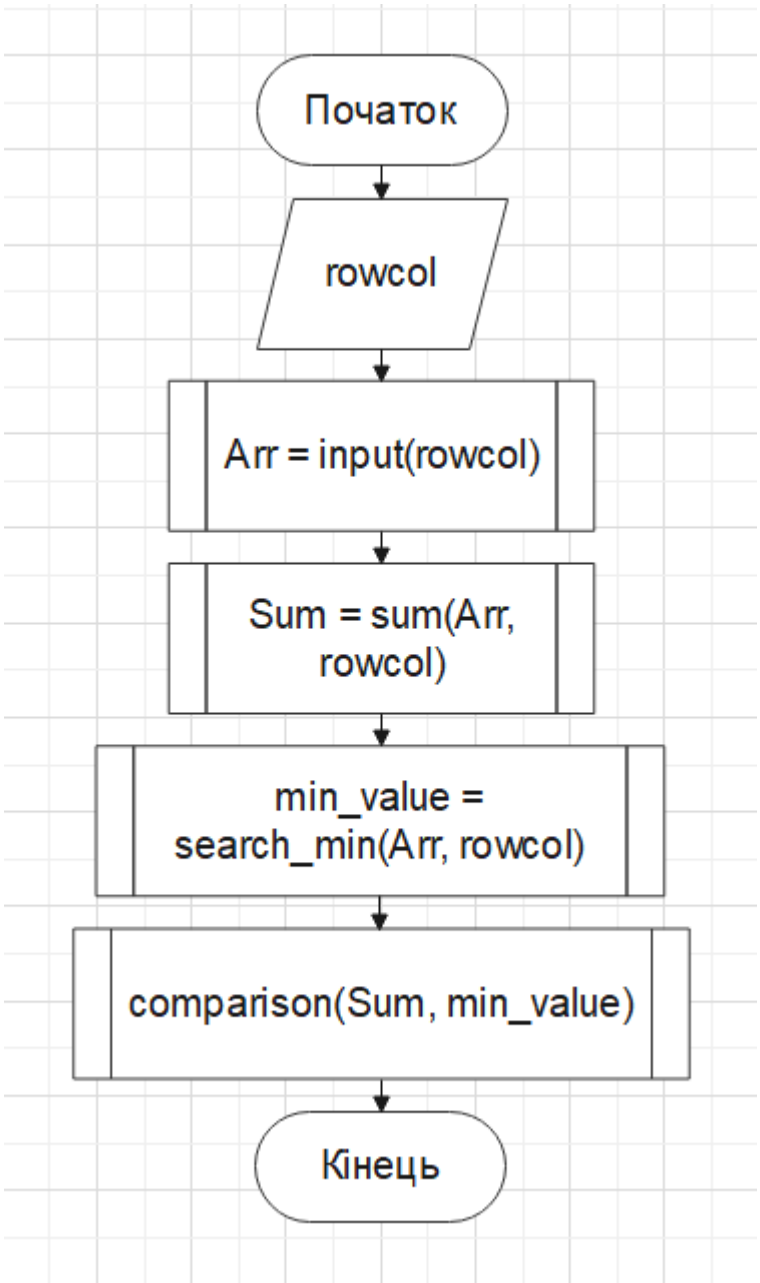
Інакше

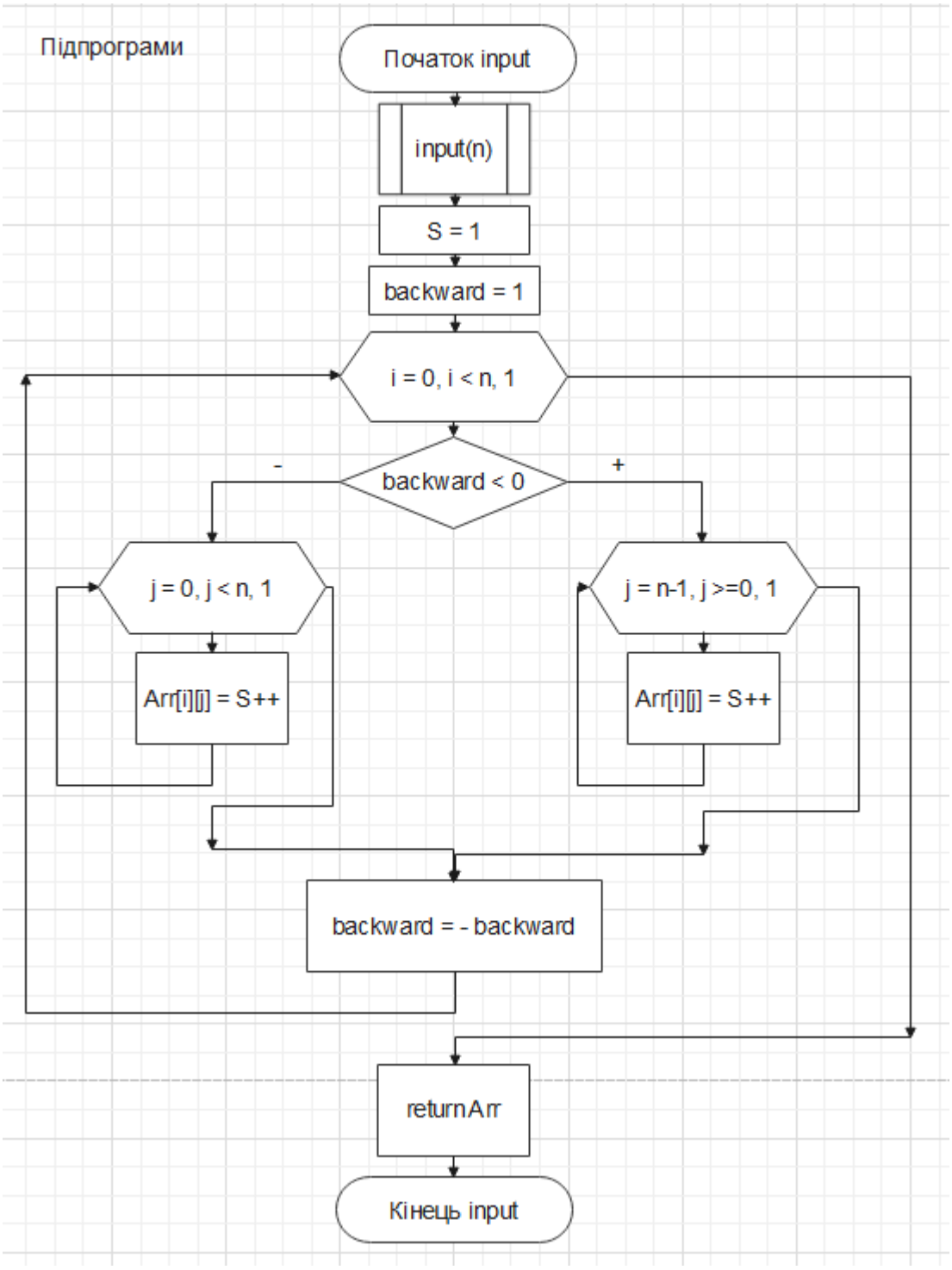
Вивід “Числа різні”

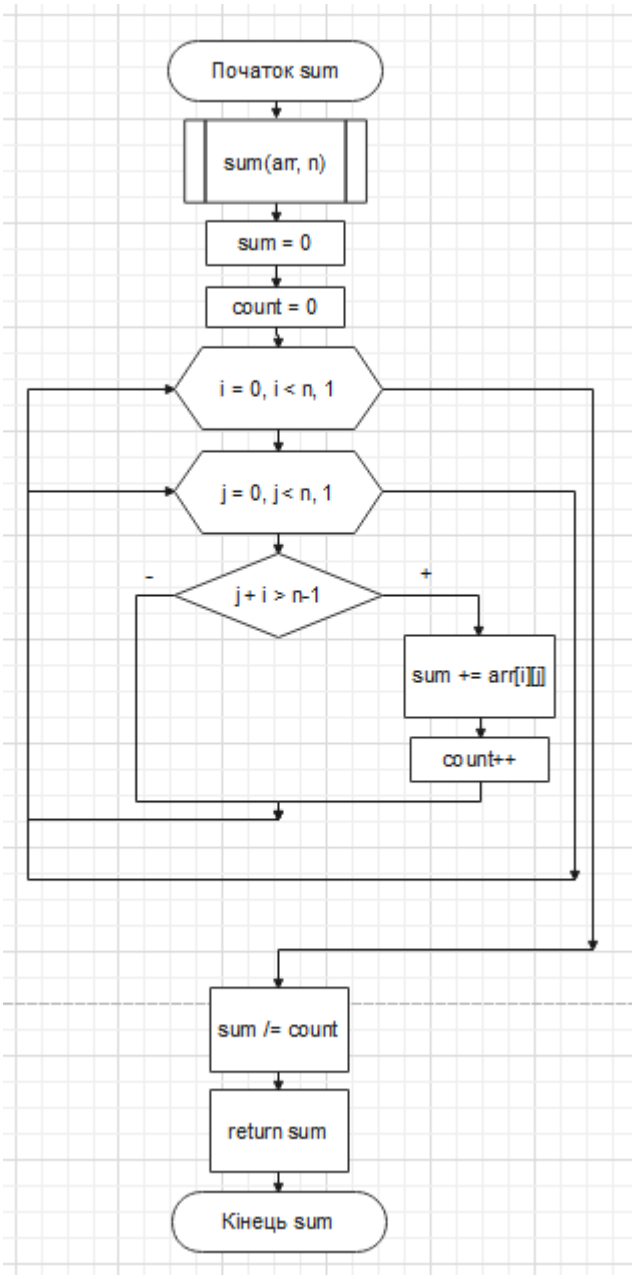
Все інакше

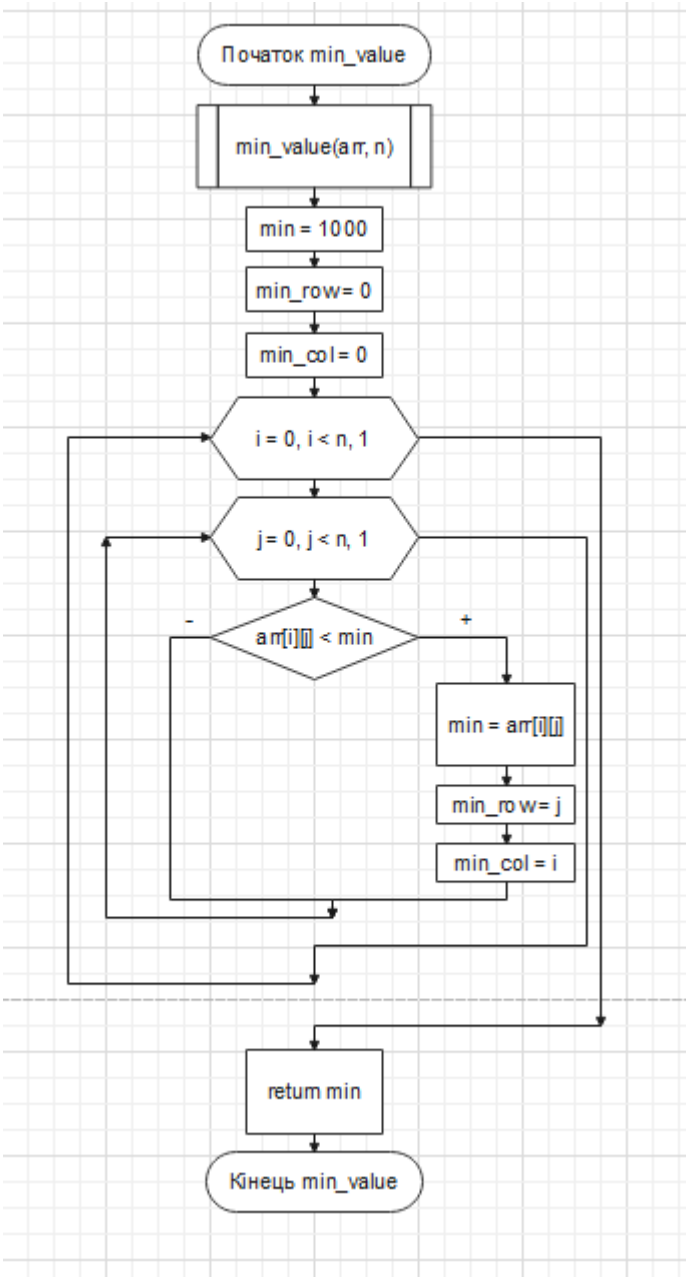
Кінець comparison

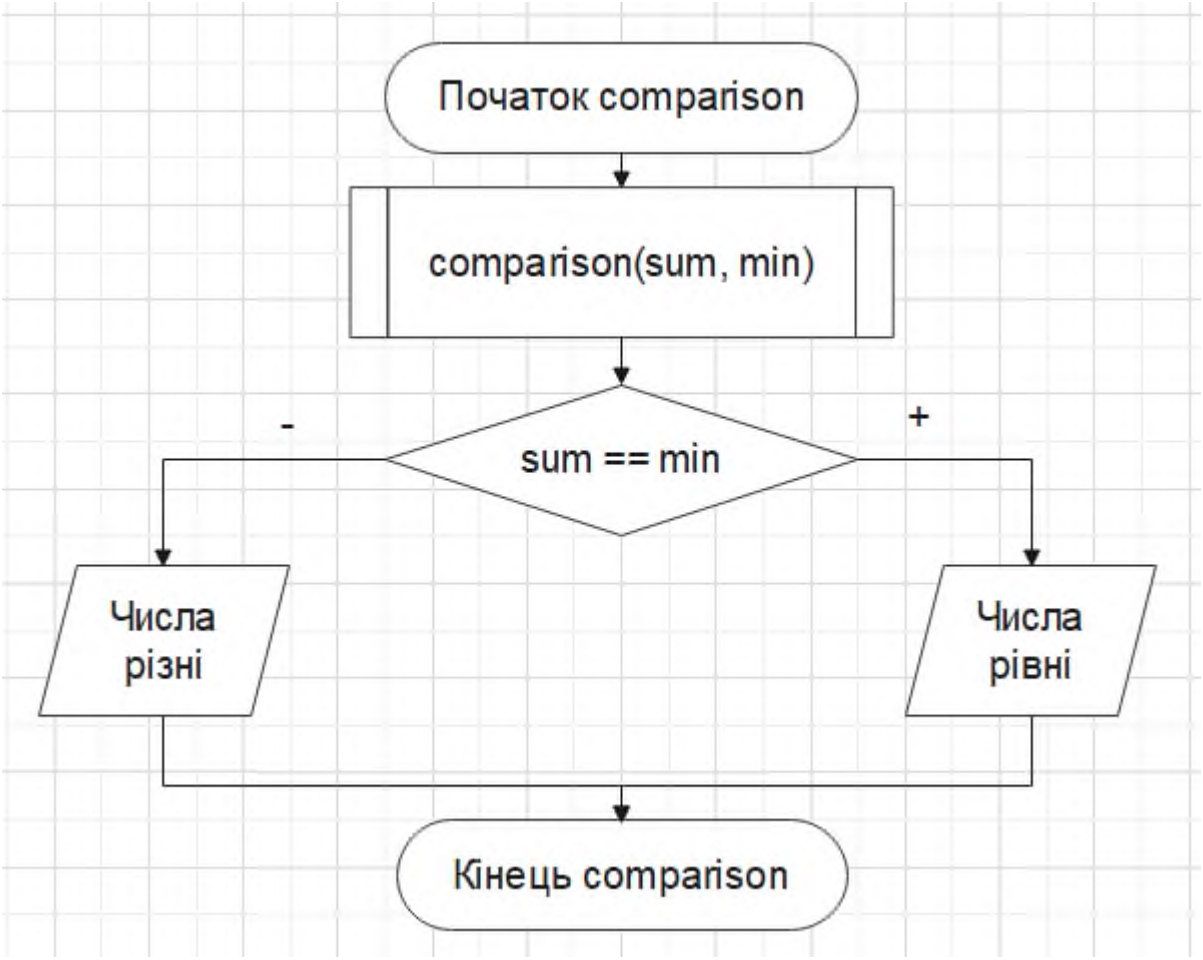
Блок-схема:


Початок
rowcol
Arr = input(rowcol)
Sum = sum(Arr, rowcol)
min_value = search_min(Arr, rowcol)
comparison(Sum, min_value)
Кінець


Підпрограми
Початок input
input(n)
S = 1
backward = 1
i = 0, i < n, 1
backward < 0
-
+
j = 0, j < n, 1
Arr[i][j] = S++
j = n-1, j >=0, 1
Arr[i][j] = S++
backward = - backward
returnArr
Кінець input


Початок sum
sum(arr, n)
sum = 0
count = 0
i = 0, i < n, 1
j = 0, j < n, 1
j + i > n-1
-
+
sum += arr[i][j]
count++
sum /= count
return sum
Кінець sum


Початок min_value
min_value(arr, n)
min = 1000
min_row= 0
min_col = 0
i = 0, i < n, 1
j = 0, j < n, 1
arr[i][j] < min
-
+
min = arr[i][j]
min_row= j
min_col = i
return min
Кінець min_value



Код програми:

```cpp
#include <iostream>
#include <iomanip>

using namespace std;

float** input(int);
void output(float**, int);
float sum(float**, int);
float search_min(float**, int);
void comparison(float, float);
void remove(float**, int);

int main() {
    float rowcol;
    float Sum;
    float min_value;
    cout << "The size of the array ";
    cin >> rowcol;
    float** Arr;
    Arr = input(rowcol);
    cout << "Two dimensional array: " << endl; output(Arr, rowcol);
    Sum = sum(Arr, rowcol);
    cout << "\nThe sum equals: " << Sum << endl;
    min_value = search_min(Arr, rowcol);
    cout << "\nMinimal value equals: " << min_value;
    comparison(Sum, min_value);
}
//Створення матриці
float** input(int n) {
    float S = 1;
    int backward = 1;
    float** Arr = new float* [n];
    for (int i = 0; i < n; i++)
        Arr[i] = new float[n];
    for (int i = 0; i < n; i++) {
        if (backward < 0) {
            for (int j = n-1; j >= 0; j--) {
                Arr[i][j] = S++;
            }
            backward = -backward;
        }
        else {
            for (int j = 0; j < n; j++) {
                Arr[i][j] = S++;
            }
            backward = -backward;
        }
    }
    return Arr;
}
//Вивід матриці
void output(float** arr, int n) {
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            cout << setw(7) << fixed << setprecision(1) << arr[i][j];
        }
        cout << endl;
    }
}
//Знаходження ариф. суми під побічною діагоналлю
```

```
61  float sum(float** arr, int n) {
62      float sum = 0,
63          count = 0;
64      for (int i = 0; i < n; i++) {
65          for (int j = 0; j < n; j++) {
66              if ((j + i) > n-1) {
67                  sum += arr[i][j];
68                  count++;
69              }
70          }
71      }
72      sum /= count;
73      return sum;
74  }
75  //Знаходження мін значення в матриці
76  float search_min(float** arr, int n) {
77      float min = 100;
78      int min_row = 0,
79          min_col = 0;
80      for (int i = 0; i < n; i++) {
81          for (int j = 0; j < n; j++) {
82              if (arr[i][j] < min) {
83                  min = arr[i][j];
84                  min_row = i;
85                  min_col = j;
86              }
87          }
88      }
89      cout << "The position: [" << min_col << "," << min_row << "]";
90      return min;
91  }
92  //Порівняння ариф. суми та мін. елемента
93  void comparison(float sum, float min) {
94      if (sum == min)
95          cout << "\nThey are equal.";
96      else
97          cout << "\nThey are different.";
98  }
99  //Видалення матриці
100 void remove(float** arr, int n) {
101     for (int i = 0; i < n; i++)
102         delete[]arr[i];
103     delete[]arr;
104 }
```

Тестування програми:

```
Microsoft Visual Studio Debug Console
The size of the array 4
Two dimensional array:
   1.0    2.0    3.0    4.0
   8.0    7.0    6.0    5.0
   9.0   10.0   11.0   12.0
  16.0   15.0   14.0   13.0

The sum equals: 11.7
The position: [0,0]
Minimal value equals: 1.0
They are different.
```

Висновок: На цій лабораторній роботі, ми дослідили алгоритми обходу масивів та набули практичних навичок використання цих алгоритмів під час складання програмних специфікацій. Ми написали програму, яка створює масив з обходом по рядкам. Потім, за умовою задачі, ми знаходили середньоарифметичну суму та мінімальне значення і його місцезнаходження. Після цього написали підпрограму, яка визначає рівність цих значень.