

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 пошуку та сортування»

Варіант 31

Виконав: студент ІП-15 Ткач Владислав Анатолійович

Перевірив: \_\_\_\_\_

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

#### Завдання

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

1. Опису змінної індексованого типу (двовимірний масив) згідно з варіантом
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

31	8 x 4	Дійсний	Із добутку значень елементів рядків двовимірного масиву. Відсортувати методом Шела за зростанням.
----	-------	---------	---

#### Постановка задачі

Результатом розв'язку є відсортований масив елементами якого є добуток елементів відповідного рядка заданого двомірного масиву.

## Побудова математичної моделі

Складємо таблицю імен змінних:

Змінна	Тип	Ім'я	Призначення
К-сть рядків масиву	Символ	n	Початкове дане
К-сть стовбців масиву	Символ	m	Початкове дане
Двомірний масив	Символ	A	Початкове дане
Одномірний масив	ціле	B	Результат
Сума елементів	ціле	mas_in	Проміжне дане
Максимальний елемент	ціле	mas_dob	Проміжне дане
Номер елементу З масиву	ціле	sort_shell	Проміжне дане
Крок для сортування Шелла	ціле	d	Проміжне дане
Значення масиву (запам'ятовування)	дійсне	znach	Проміжне дане
Вивід масиву	Функція	mas_out1()	Вивід даних
Вивід двомірного масиву	Функція	mas_out2()	Вивід даних
Випадкове значення	функція	rand()	Обрахунок даних
Зміна типу	функція	double()	Обрахунок даних
Максимальне випадкове значення	ціле	RAND_MAX	Проміжне дане

Для визначення результату зробимо наступні дії:

Ініціалізуємо двомірний масив випадковими значеннями використовуючи функцію **mas\_in()**, у якій використовуємо формулу для визначення випадкового дійсного числа від 0 до 10: **(double)(rand()) / RAND\_MAX \* 10**

Знайдемо добуток елементів кожного рядка двомірного масиву та запишемо ці дані в одномірний використовуючи функцію **mas\_dob()**, де sum перед початком другого масиву рівне завжди 1, для правильного запису.

Потім за допомогою функції **sort\_shell()** відсортуємо масив за методом Шелла

Для зручності також використаємо функції виведення масиву:

**mas\_out1()** – виведення одномірного масиву

**mas\_out2()** – виведення двомірного масиву

## Розв'язання

Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо дію ініціювання двомірного масиву

Крок 3. Деталізуємо дію визначення одномірного масиву

Крок 4. Деталізуємо дію сортування масиву методом Шелла

### **Псевдокод алгоритму**

#### **Крок 1**

##### **Початок**

Ініціювання двомірного масиву

Визначення одномірного масиву

Сортування масиву методом Шелла

##### **Кінець**

#### **Крок 2**

##### **Початок**

$n = 8$

$m = 4$

`mas_in(A, n, m)`

Визначення одномірного масиву

Сортування масиву методом Шелла

##### **Кінець**

### **Крок 3**

#### **Початок**

$n = 8$

$m = 4$

`mas_in(A, n, m)`

`mas_out2(A, n, m)`

`mas_dob(a, B, n, m)`

Сортування масиву методом Шелла

#### **Кінець**

### **Крок 4**

#### **Початок**

$n = 8$

$m = 4$

`mas_in(A, n, m)`

`mas_out2(A, n, m)`

`mas_dob(a, b, n, m)`

`sort_shell(b, n)`

`mas_out1(b, n)`

#### **Кінець**

## Підпрограми:

**mas\_in(mas[], n, m)**

**для i < n повторити**

**для j < m повторити**

mas[i][j] = (double)(rand()) / RAND\_MAX \* 10

**все повторити**

mas2[i] = sum

**все повторити**

**mas\_dob(mas[] mas2[], n, m)**

**для i < n повторити**

sum = 1

**для j < m повторити**

sum = sum \* mas[i][j]

**все повторити**

mas2[i] = sum

**все повторити**

**sort\_shell(mas[], n)**

$d = n / 2$

**поки**  $d > 0$  **повторити**

**для**  $i < n$  **повторити**

$j = i$

**поки**  $j > 0 \ \&\& mas[i] > mas[j + d]$  **повторити**

$znach = mas[j]$

$mas[j] = mas[j + d]$

$mas[j + d] = znach$

$j = j - 1$

**все повторити**

**все повторити**

$d = d / 2$

**все повторити**

**mas\_out1 (mas[], n)**

**для**  $i < n$  **повторити**

**вивід**  $mas[i]$

**все повторити**

**mas\_out2(mas[], n, m)**

**для i < n повторити**

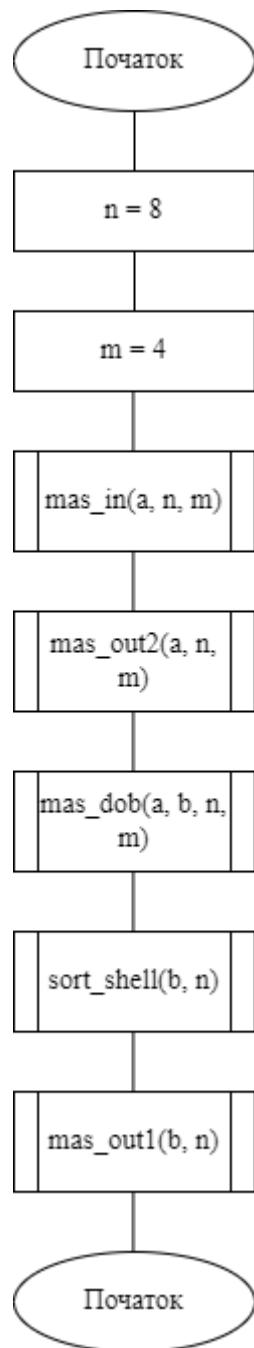
**для j < m повторити**

**вивід mas[i][j]**

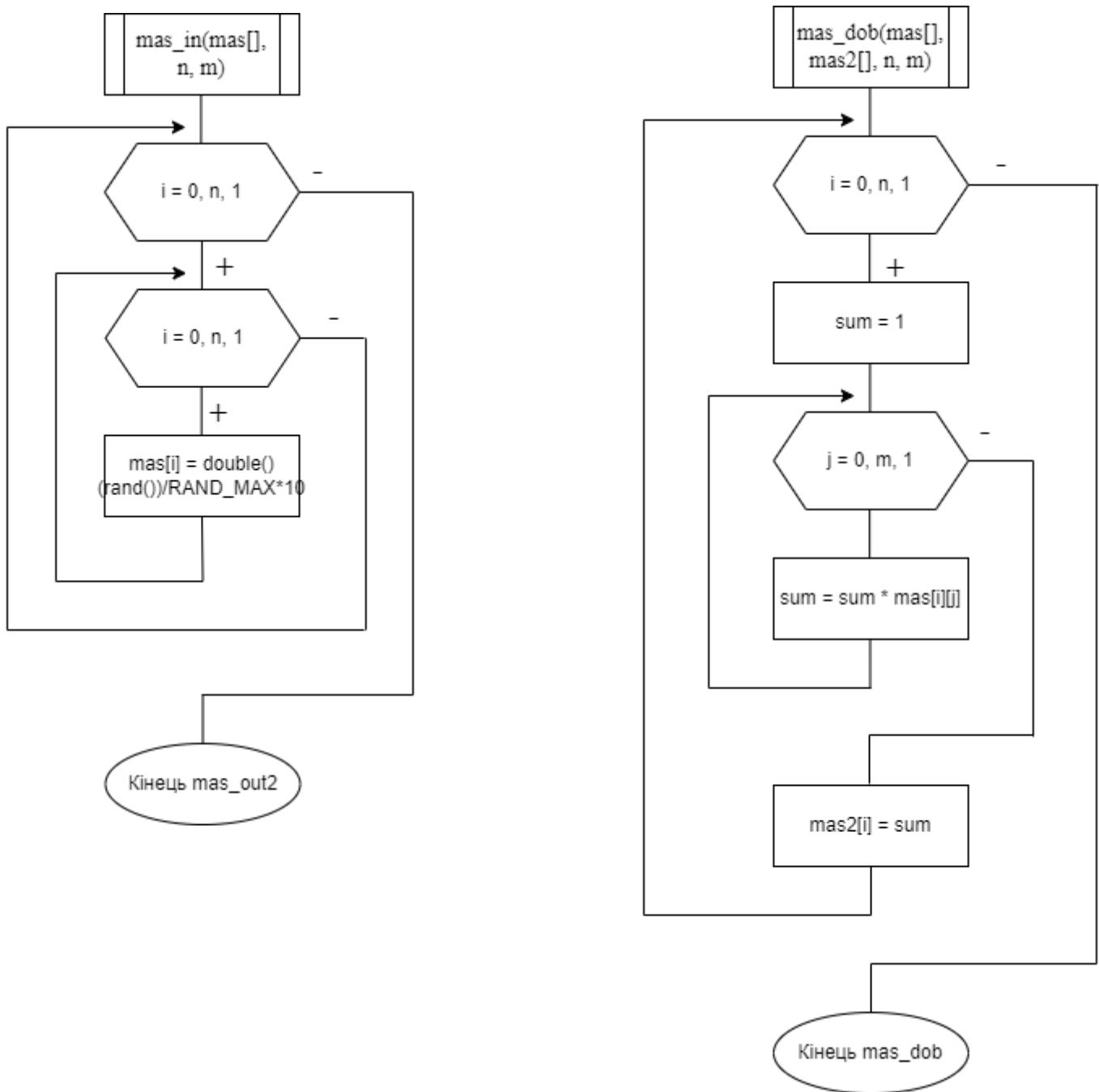
**все повторити**

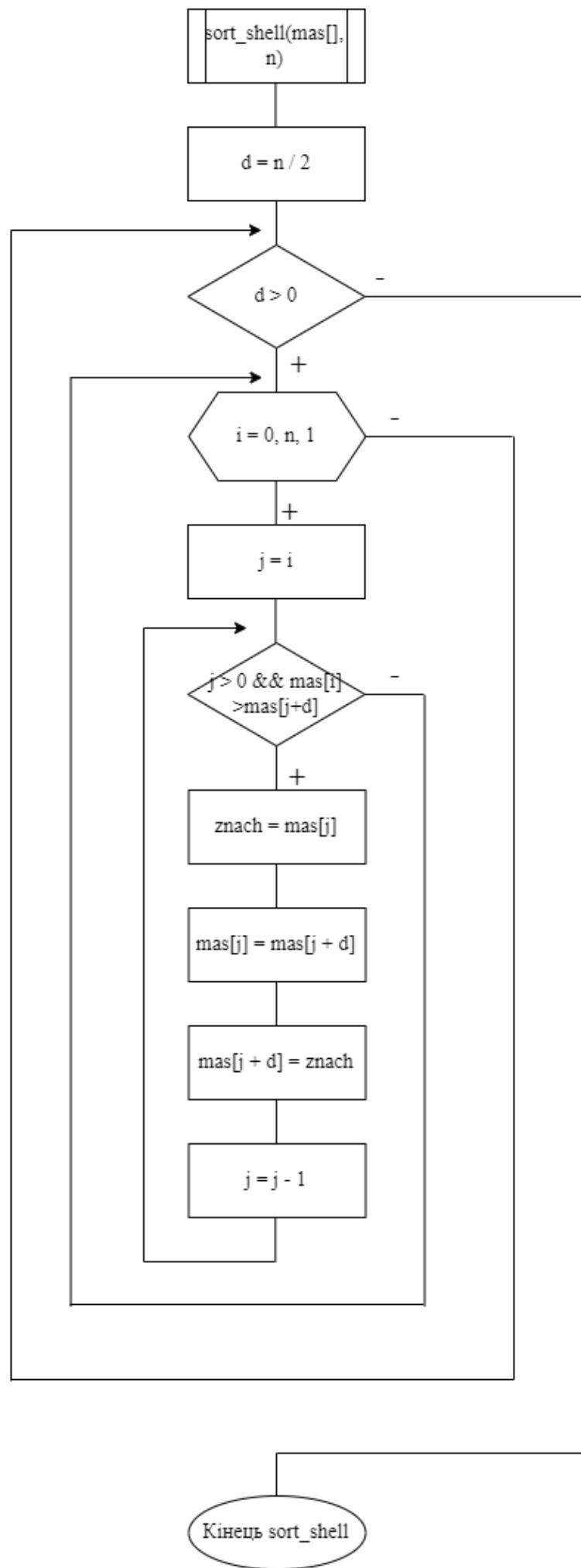
**все повторити**

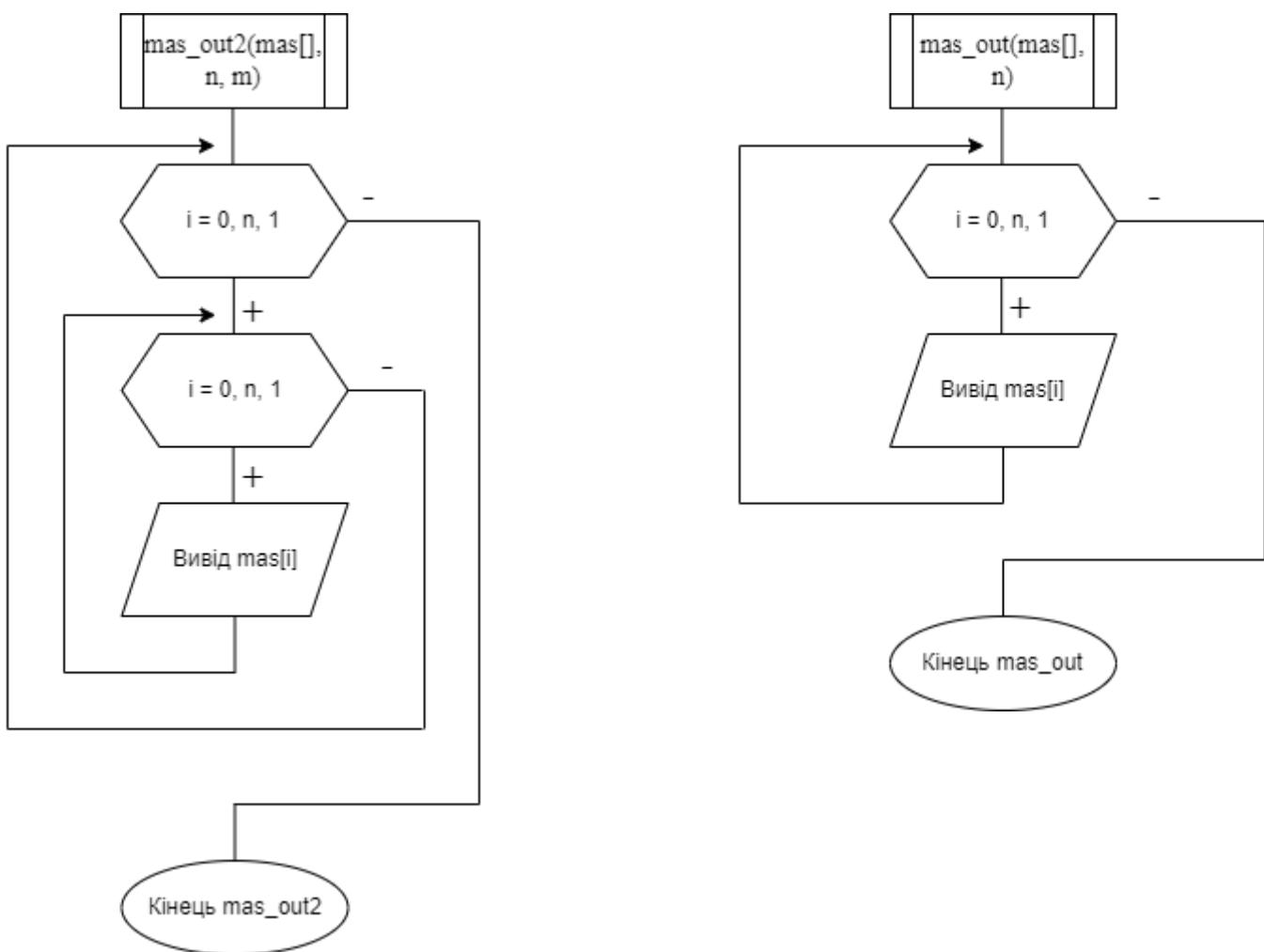
### Блок схема алгоритму



## Підпрограми







### Код програми

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
void mas_in( double *mas[], int n, int m){
```

```
    srand(time(NULL));
```

```
    for(int i = 0; i < n; i++){
```

```
        for(int j = 0; j < m; j++){
```

```
            mas[i][j] = (double)(rand()) / RAND_MAX * 10;
```

```
    }
```

```
}
```

```
}
```

```
void mas_out1( double mas[], int n){
```

```
    for(int i = 0; i < n; i++){
```

```
        cout << fixed << setprecision(2) << mas[i] << " ";
```

```
    }
```

```
    cout << endl;
```

```
}
```

```
void mas_out2( double *mas[], int n, int m){
```

```
    for(int i = 0; i < n; i++){
```

```
        for(int j = 0; j < m; j++){
```

```
cout << fixed << setprecision(2) << mas[i][j] << " ";
```

```
}
```

```
cout << endl;
```

```
}
```

```
}
```

```
void mas_dob( double *mas[], double mas2[], int n, int m){
```

```
    double sum;
```

```
    for(int i = 0; i < n; i++){
```

```
        sum = 1;
```

```
        for(int j = 0; j < m; j++){
```

```
            sum *= mas[i][j];
```

```
        }
```

```
        mas2[i] = sum;
```

```
    }
```

```
}
```

```
void sort_shell(double mas[], int n) {
```

```
    int d = n / 2;
```

```
    while (d > 0) {
```

```
        for (int i = 0; i < n - d; i++) {
```

```
            int j=i;
```

```
while (j >= 0 && mas[j] > mas[j + d]) {  
    double znach = mas[j];  
    mas[j] = mas[j + d];  
    mas[j + d] = znach;  
    j--;  
}  
}  
d = d / 2;  
}
```

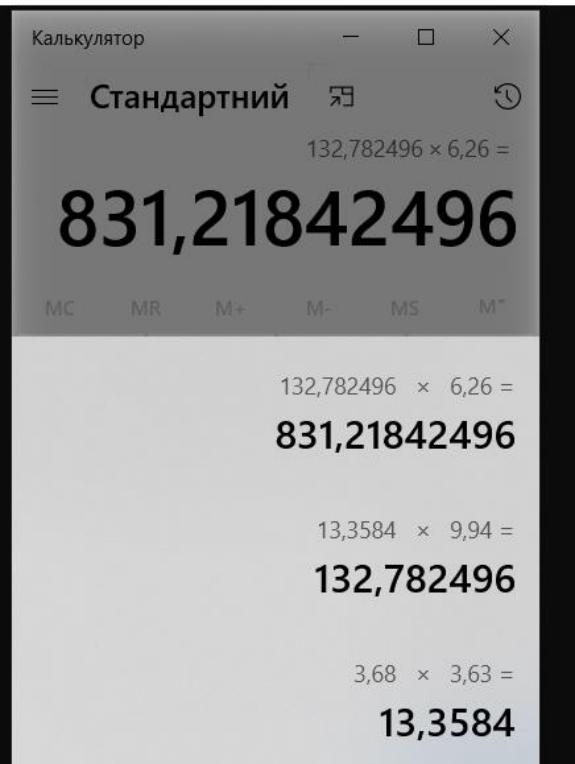
```
int main(){  
    int n = 8 , m = 4;  
    double A[n][m];  
    double *a[n];  
  
    for(int i = 0; i < n; i++){  
        a[i] = &A[i][0];  
    }  
  
    mas_in(a, n, m);  
    cout << "MAS: " << endl;  
    mas_out2(a, n, m);
```

```
double B[n];  
  
mas_dob(a, B, n, m);  
  
cout << "not sort mas: " << endl;  
  
mas_out1(B, n);  
  
sort_shell(B, n);  
  
cout << "sort mas: " << endl;  
  
mas_out1(B, n);  
  
}
```

## Випробування програми

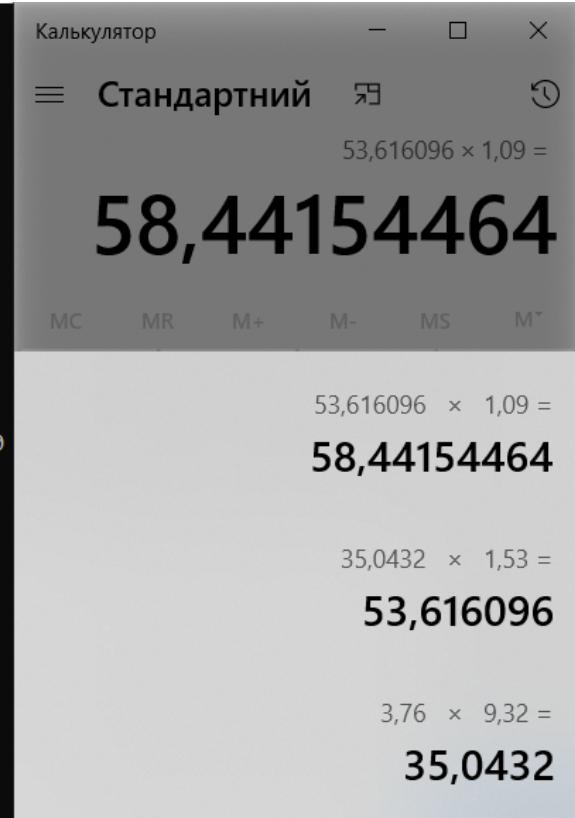
```
MAS:
3.68 3.63 9.94 6.26
5.19 8.80 9.62 8.71
2.27 7.78 1.67 6.72
2.37 5.11 6.75 0.45
3.74 4.88 1.04 1.02
9.01 6.25 0.90 2.85
9.42 7.05 3.29 8.61
1.20 0.78 6.10 4.59
not sort mas:
830.82 3824.77 198.39 36.71 19.34 144.20 1884.23 26.30
sort mas:
19.34 26.30 36.71 144.20 198.39 830.82 1884.23 3824.77

-----
Process exited after 0.04243 seconds with return value 0
Press any key to continue . . .
```



```
MAS:
3.76 9.32 1.53 1.09
0.16 5.87 6.56 6.35
0.42 5.80 8.02 7.81
9.62 4.12 5.33 9.03
6.66 0.33 6.30 2.53
1.20 8.92 2.05 6.24
0.40 0.42 0.02 5.88
8.22 4.69 0.28 7.07
not sort mas:
58.37 38.08 153.38 1907.48 35.50 137.00 0.02 76.75
sort mas:
0.02 35.50 38.08 58.37 76.75 137.00 153.38 1907.48

-----
Process exited after 0.04446 seconds with return value 0
Press any key to continue . . .
```



## Висновки

Ми дослідили методи Дослідження алгоритмів пошуку та сортування та набули практичних навичок їх використання під час складання програмних специфікацій.

У результаті виконання лабораторної роботи ми отримали алгоритм для визначення відсортованого масиву методом Шелла, у якому елементи масиву це добуток елементів рядків двомірного масиву, : визначили основні дії, деталізували дію ініціювання двомірного масиву, дію визначення одномірного масиву та дію сортування масиву методом Шелла.