Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний

інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженері

Звіт

з лабораторної роботи № 8 з дисципліни

«Алгоритми та структури даних-1.

Основи алгоритмізації»

Дослідження алгоритмів пошуку та сортування

Варіант 31

Виконав студент ІП-12 Титаренко Данило Олегович

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

**Задача 31**

Дано дійсний масив розміром 8 х 4. Із добутку елементів рядків двовимірного масиву створити одновимірний масив та відсортувати методом Шела за зростанням.

**Розв’язок**

1. **Постановка задачі.**

За допомогою вкладених циклів ініціалізуємо двовимірний масив. Потім знаходимо добутки елементів рядків та заповнюємо ними ж одновимірний масив. Потім використовуємо алгоритм для методу Шела. Результатом виведення буде одновимірний масив.

1. **Математична модель.** Складемо таблицю імен змінних

|  |  |  |  |
| --- | --- | --- | --- |
| Змінна | Тип | Ім’я | Призначення |
| Двовимірний масив | Дійсний | a | Початкове дане |
| Одновимірний масив | Дійсний | с | Результат |
| К-ть рядків | Цілий | n | Початкове дане |
| К-ть стовпчиків | Цілий | m | Початкове дане |
| Добуток елементів | Дійсний | d | Проміжне дане |
| Лічильник 1 | Цілий | i | Проміжне дане |
| Лічильник 2 | Цілий | j | Проміжне дане |
| Лічильник 3 | Цілий | gap | Проміжне дане |
| Допоміжна змінна | Цілий | t | Проміжне дане |

Таким чином математичне формулювання задачі зводиться до ініціалізування двовимірного масиву за допомогою вкладених ітераційних циклів, знаходження добутків елементів рядків також за допомогою ітераційних циклів, а також застосування методу Шела для сортування масиву і виведення одновимірного масиву також за допомогою ітераційного циклу.

1. **Розв’язок**

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

*Крок 1.* Визначимо основні дії

*Крок 2*. Ініціалізуємо елементи двовимірного масиву

*Крок 3.*. Деталізуємо дію перебору рядків множини послідовностей

*Крок 4.* Деталізуємо дію обчислення добуток елементів рядка

*Крок 5.* Деталізуємо дію сортування масиву за допомогою методу Шела за зростанням

*Крок 6*. Деталізуємо дію виводу одновимірного масиву

1. **Псевдокод**

Крок 1

**Початок**

Ініціалізуємо елементи двовимірного масиву

Деталізуємо дію перебору рядків множини послідовностей

Деталізуємо дію обчислення добуток елементів рядка

Деталізуємо дію сортування масиву за допомогою методу Шела за зростанням

Деталізуємо дію виводу одновимірного масиву

**Кінець**

Крок 2

**Початок**

**повторити**

**для і від 0 до n**

**повторити**

**для j від 0 до m**

a[i][j]=rand()%20+1;

**все повторити**

**все повторити**

Деталізуємо дію перебору рядків множини послідовностей

Деталізуємо дію обчислення добуток елементів рядка

Деталізуємо дію сортування масиву за допомогою методу Шела за зростанням

Деталізуємо дію виводу одновимірного масиву

**Кінець**

Крок 3

**Початок**

**повторити**

**для і від 0 до n**

**повторити**

**для j від 0 до m**

a[i][j]=rand()%20+1;

**все повторити**

**все повторити**

**повторити**

**для і від 0 до n**

d=1

Деталізуємо дію обчислення добуток елементів рядка

Деталізуємо дію сортування масиву за допомогою методу Шела за зростанням

Деталізуємо дію виводу одновимірного масиву

**Кінець**

Крок 4

**Початок**

**повторити**

**для і від 0 до n**

**повторити**

**для j від 0 до m**

a[i][j]=rand()%20+1;

**все повторити**

**все повторити**

**повторити**

**для і від 0 до n**

d=1

**повторити**

**для j від 0 до m**

d = d \* a[i][j];

**все повторити**

c[i] = d;

**все повторити**

Деталізуємо дію сортування масиву за допомогою методу Шела за зростанням

Деталізуємо дію виводу одновимірного масиву

**Кінець**

Крок 5

**Початок**

**повторити**

**для і від 0 до n**

**повторити**

**для j від 0 до m**

a[i][j]=rand()%20+1;

**все повторити**

**все повторити**

**повторити**

**для і від 0 до n**

d=1

**повторити**

**для j від 0 до m**

d = d \* a[i][j];

**все повторити**

c[i] = d;

**все повторити**

**повторити**

**для gap від n/2 до 0 з кроком gap/2**

**повторити**

**для і від gap до n з кроком 1**

**повторити**

**для j від (i-gap)до ((j >= 0 && c[j] > c[j + gap]) з кроком -gap**

t = c[j];

c[j] = c[j + gap];

c[j + gap] = t;

**все повторити**

**все повторити**

**все повторити**

Деталізуємо дію виводу одновимірного масиву

**Кінець**

Крок 6

**Початок**

**повторити**

**для і від 0 до n**

**повторити**

**для j від 0 до m**

a[i][j]=rand()%20+1;

**все повторити**

**все повторити**

**повторити**

**для і від 0 до n**

d=1

**повторити**

**для j від 0 до m**

d = d \* a[i][j];

**все повторити**

c[i] = d;

**все повторити**

**повторити**

**для gap від n/2 до 0 з кроком gap/2**

**повторити**

**для і від gap до n з кроком 1**

**повторити**

**для j від (i-gap)до ((j >= 0 && c[j] > c[j + gap]) з кроком -gap**

t = c[j];

c[j] = c[j + gap];

c[j + gap] = t;

**все повторити**

**все повторити**

**все повторити**

**повторити**

**для і від 0 до n**

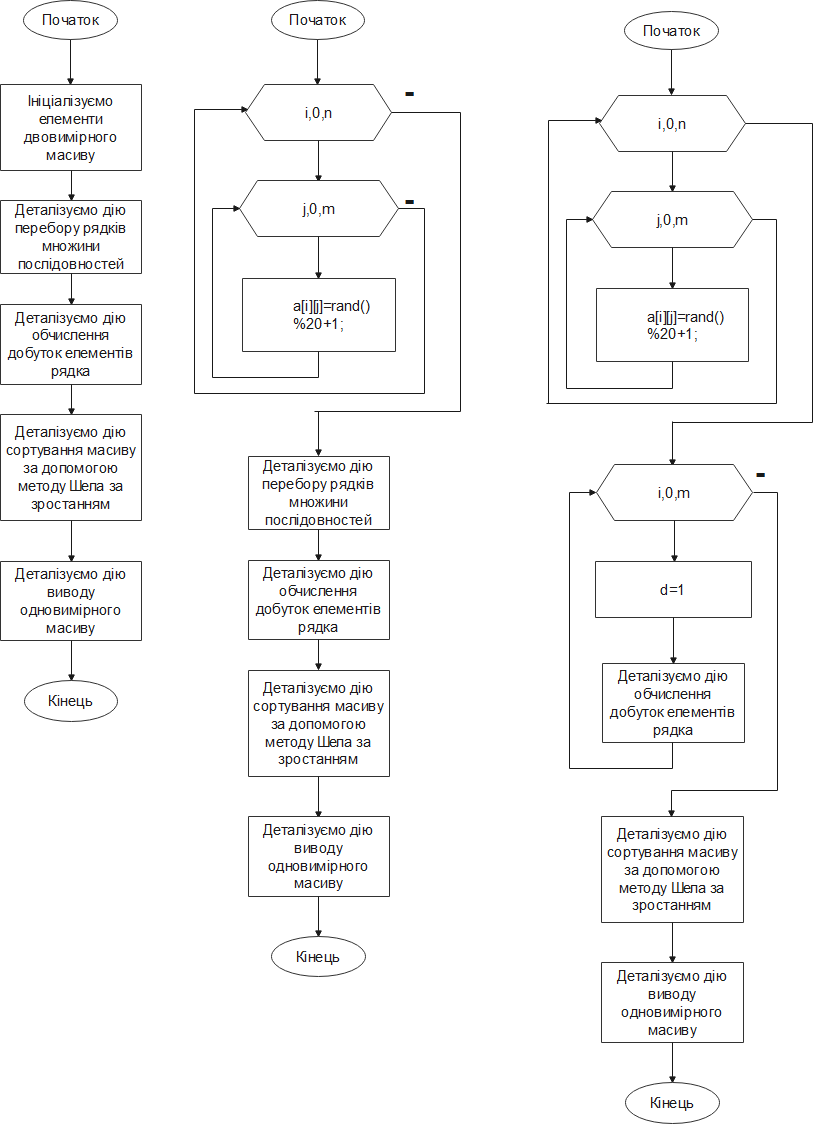
**Вивід с[i]**

**все повторити**

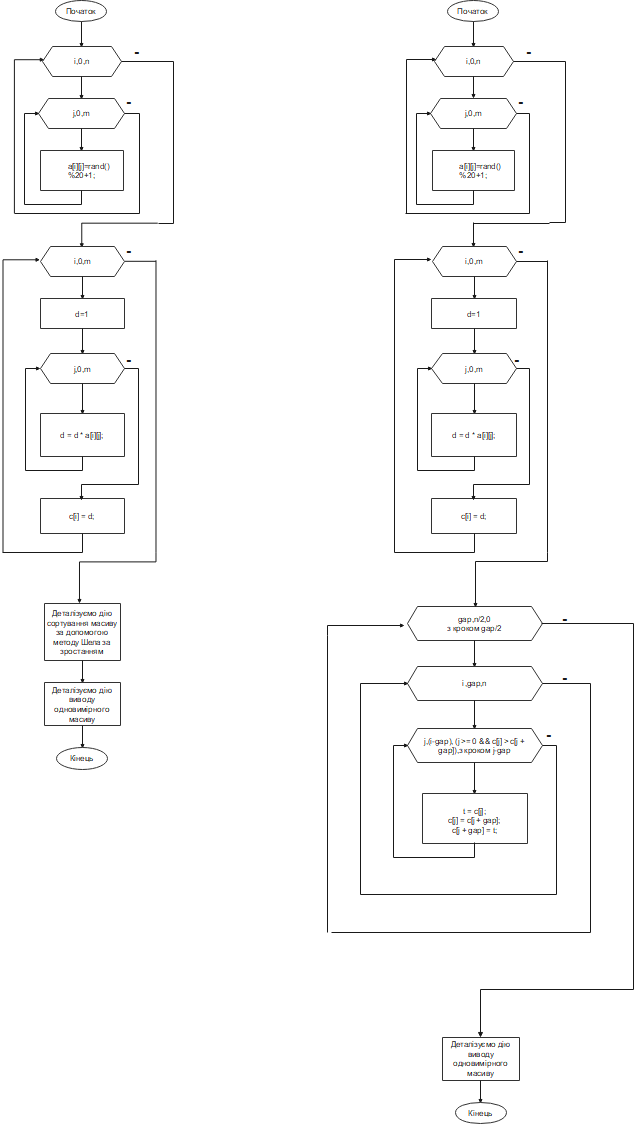
**Кінець**

1. **Блок-схема**

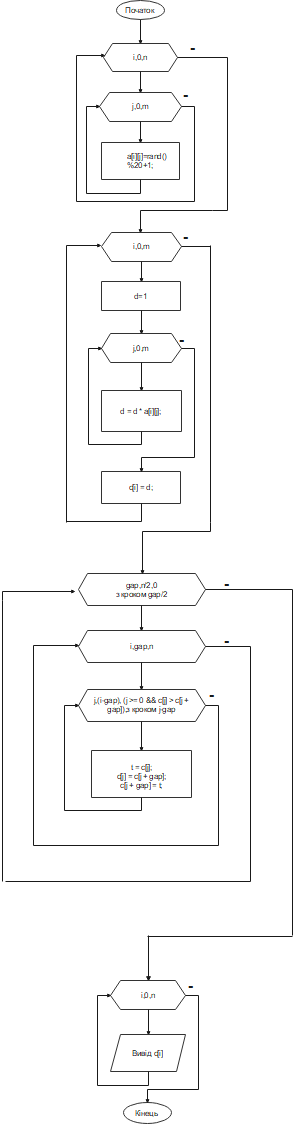
Крок 1 Крок 2 Крок 3



Крок 4 Крок 5



Крок 6



1. **Код програми**

#include <iostream>

#include <iomanip>

void input(float\*\* a, int n, int m) {

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

a[i] = new float[m];

}

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

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

a[i][j] = rand() % 20+1;

}

}

}

void output(float\*\* a, int n, int m) {

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

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

std::cout << std::setw(2) << a[i][j] << " ";

}

std::cout << std::endl;

}

}

int main()

{

srand(time(NULL));

int n, m;

std::cout << "input n,m";

std::cin >> n;

std::cin >> m;

float\*\* a = new float\* [n];

float\* c = new float[n];

std::cout << std::endl;

input(a, n, m);

output(a, n, m);

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

float d = 1.F;

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

d = d \* a[i][j];

}

c[i] = d;

}

std::cout << std::endl;

float t;

for (int gap = n / 2; gap > 0; gap /= 2)

{

for (int i = gap; i < n; i++)

{

for (int j = i - gap; (j >= 0 && c[j] > c[j + gap]); j -= gap)

{

t = c[j];

c[j] = c[j + gap];

c[j + gap] = t;

}

}

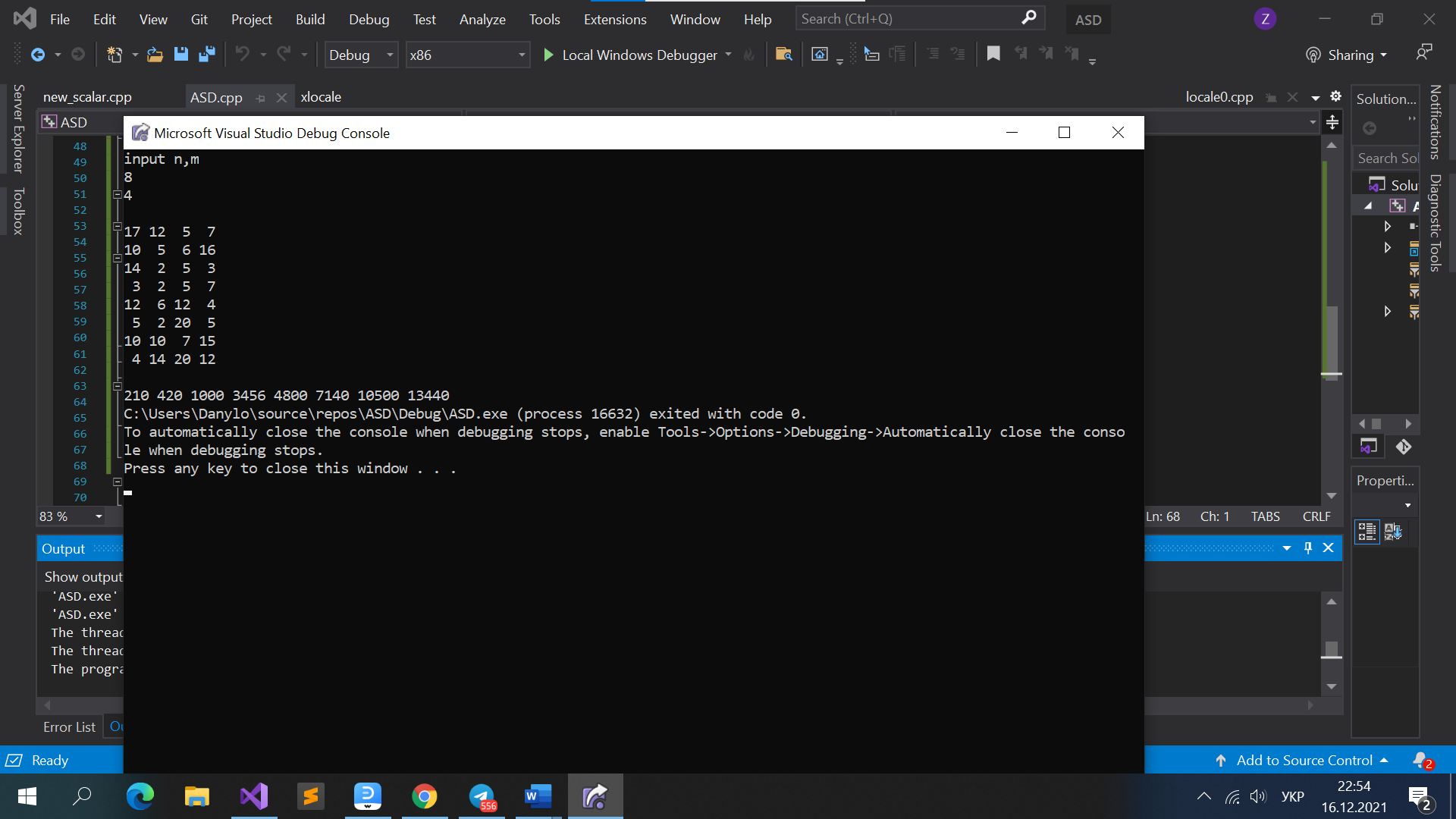
}

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

std::cout << c[i] << " ";

}

}



**7. Випробовування алгоритму**

Перевіримо правильність алгоритму на довільних конкретних значеннях початкових даних.

n=8 m=4

|  |  |
| --- | --- |
| Блок | Дія |
|  | Початок |
| 1 | i=1; 1<=10;true  a[1]=120-1=119  i=1+1=2  …  a[10]=120-10=110  i=10+1=11  i=11; 11<=10;false  a[10]=[119,118,117,116,115,114,113,112,111,110] |
| 2 | j=1; 1<=10;true  b[1]=110+1=111  j=1+1=2  …  b[10]=110+10=110  j=10+1=11  j=11; 11<=10;false  b[10]=[111,112,113,114,115,116,117,118,119,120] |
| 3 | i=1; 1<=10;true  j=1; 1<=10;true  119=111; false  J=1+1=2;  J=2;2<=10; true  …  119=119;true  N=0+1=1  C[1]=119  J=9+1=10  …  j=11; 11<=10;false  i=1+1=2  …  i=11; 11<=10;false  c[9]=[119,118,117,116,115,114,113,112,111]  n=9 |
| 4 | I=1  S=0  Max=0  I=1;1<=9;true  S=0+119=119  119>0;true  Max=119  I=i+1=1+1=2  …  I=10;10<=11;false  S=1035  Max=119  S=1035/9=115 |
| 5 | I=1  I=1;i<=9;true  C[1]=max=119  C[1]=115  I=1+1=2  …  I=10;10<=9;false  C[8]=[210,420,1000,3456,7140,10500,13440] |
| 6 | I=0  I=0;1<8;true  Вивід c[i]  I=1+1=2  …  I=9;9<8; false |
|  | Кінець |

1. **Висновок**

Було досліджено алгоритми пошуку та сортування, набуто практичних навичок використання цих алгоритмів під час складання програмних специфікацій. Особливістю моєї роботи було застосування методу Шелла для сортування масиву за зростанням. Також декомпозував задачу на 6 кроків: визначив основні дії, ініціалізував двовимірний масив випадковими числами, перебрав рядки множини послідовності, знайшов добутки елементів рядків та ініціалізував ними же одновимірний масив, сортував масив за методом Шелла за зростанням та вивів відсортований масив дійсних чисел.