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 30

Виконав студент ІП-12 Тарасюк Євгеній Сергійович

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.

Задача 30.

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. Створення нової змінної індексованого типу (одновимірний масив) та її ініціювання значеннями, що обчислюються згідно з варіантом (табл. 1).

30	7 x 6	Дійсний	Із добутку від'ємних значень елементів рядків двовимірного масиву. Відсортувати методом Шела за спаданням.

Розв'язок.

1. **Постановка задачі.**

Програма працює без вхідних даних. Для ініціювання змінних та обчислень використовуватимемо арифметичні цикли та цикли з передумовою. Результатом розв'язку є два масиви чисел. Використовуватимемо стандартні логічні та арифметичні операції, функцію "randFloat(a, b)", де [a; b] проміжок, у якому генерується випадкове дійсне число.

2. Побудова математичної моделі

Таблиця змінних та функцій:

Змінні та функції	Тип	Ім’я	Призначення
висота	**Константа = 7**	**height**	
ширина	**Константа = 6**	**width**	
Двовимірний масив	**Двовимірний масив дійсних чисел**	**array2D**	**Двовимірний масив**
Одновимірний масив	**Одновимірний масив дійсних чисел**	**array1D**	**Одновимірний масив**
i	**Ціле число**	**i**	**Змінна для перелічування рядків**
j	**Ціле число**	**j**	**Змінна для перелічування стовпців**
Заповнення 2D	**Функція, не повертає значення**	***fill2D(****array2D****)***	**Заповнює array2D випадковими числами**
Заповнення 1D	**Функція, не повертає значення**	***fill1D(****array2D, array1D****)***	**Заповнює масив добутками від’ємних елементів рядків array2D**
Добуток	**Дійсне число**	***prod***	**Добуток елементів**
Сортування Шела	**Функція, не повертає значення**	***shellSort(****array1D****)***	**Сортування масиву методом Шела**
temp	**Дійсне число**	***temp***	**Допоміжна змінна для того, щоб**

			міняти елементи масиву місцями
d	**Натуральне число**	***d***	**Дистанція між елементами, що порівнюються**
Виведення 1D	**Функція, не повертає значення**	***print1D(****array1D****)***	**Виводить одновимірний масив**
Виведення 2D	**Функція, не повертає значення**	***print2D(****array2D****)***	**Виводить двовимірний масив**
Випадкове число	**Функція, повертає дійсне число**	***randFloat(a, b)***	**Генерує випадкове число на проміжку [a; b]**

3. Псевдокод алгоритму

Початок

fill2D(*array2D)*

fill1D(*array2D, array1D)*

print2D*(array2D)*

print1D*(array1D)*

shellSort*(array1D)*

print1D*(array1D)*

Кінець.

Функція *fill2D(**array2D****)***

Початок

Повторити для *i* **на проміжку** *[0; height)*

Повторити для *j* **на проміжку** *[0; width)*

array2D[i][j] **= randFloat**(-50, 50)

Все повторити

Все повторити

Повернення

Кінець

Функція *fill1D(**array2D, array1D****)***

Початок

Повторити для *i* **на проміжку** *[0; height)*

prod = 1

Повторити для на j проміжку *[0; width)*

Якщо *array2D[i][j] < 0*

*prod *= array2D[i][j]*

Все якщо

Все повторити

array1D[i] = prod

` **Все повторити**

Повернення

Кінець

Функція *shellSort(**array1D****)***

Початок

d = height / 2

Повторити поки *d >= 1*

Повторити для i **на проміжку** *[0; height)*

j = i

Повторити поки *j >= d* **та** *array1D[j-d] < array1D[j]*

temp = array1D[j]

array1D[j] = array1D[j-d]

array1D[j-d] = temp

j -= d

Все повторити

Все повторити

d /= 2;

Все повторити

Повернення

Кінець

Функція *print1D(**array1D****)***

Початок

Повторити для *i* **на проміжку** *[0; height)*

Виведення *array1D[i]*

Все повторити

Повернення

Кінець

Функція *print2D(**array2D****)***

Початок

Повторити для *i* **на проміжку** *[0; height)*

Повторити для *j* **на проміжку** *[0; width)*

Виведення *array2D[i][j]*

Все повторити

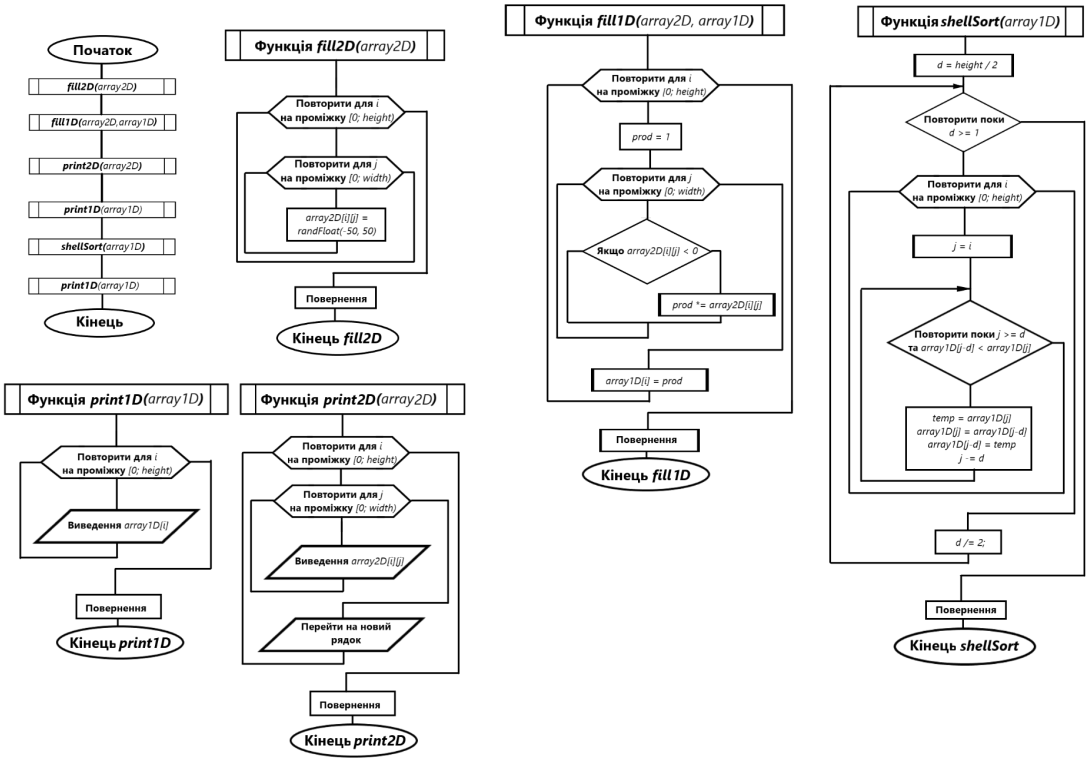
Перейти на новий рядок

Все повторити

Повернення

Кінець

4. Блок-схема алгоритму



5. Код програми (C++)

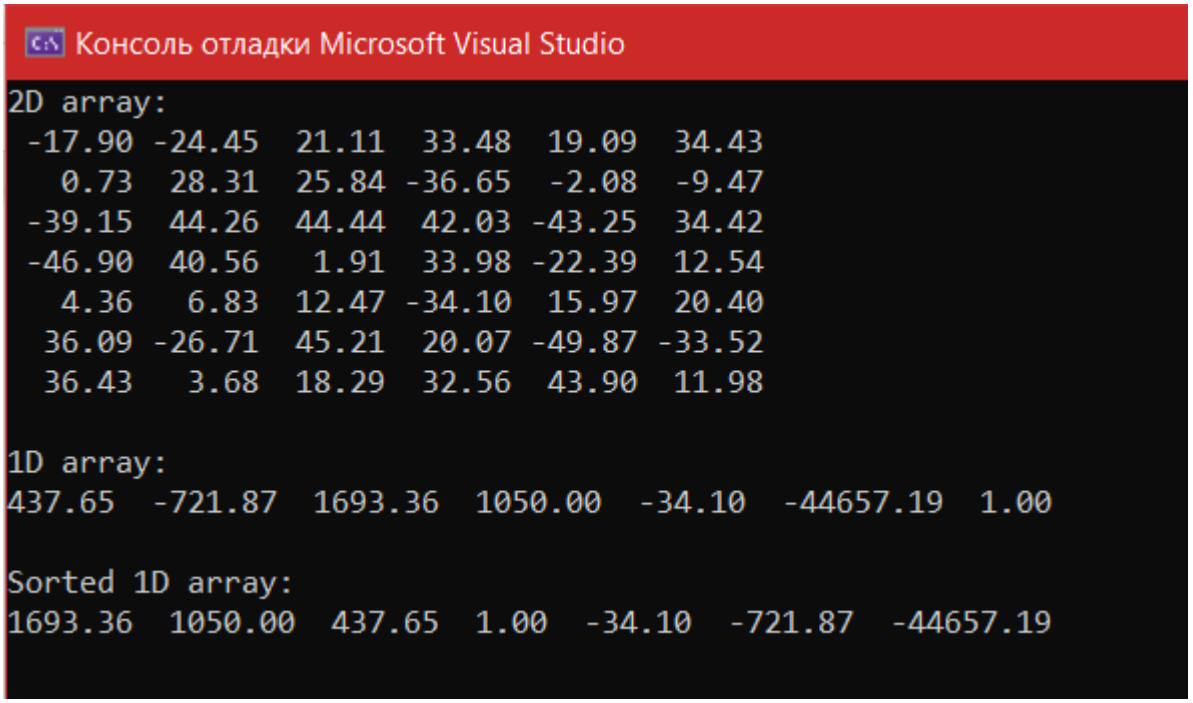
```
#include <iostream>
#include <random>
#define width 6
#define height 7
#define range 100;
#define halfRange 50;

using namespace std;

void fill2D(float**);
void fill1D(float**, float*);
void shellSort(float*);
void print1D(float*);
void print2D(float**);


int main()
{
   float** array2D = new float*[height];
   for (int i = 0; i < height; i++)
   {
      array2D[i] = new float[width];
   }
   fill2D(array2D);
   float* array1D = new float[height];
   fill1D(array2D, array1D);
   cout << "2D array: " << endl;
```

```
    print2D(array2D);
    cout << "1D array: " << endl;
    print1D(array1D);
    shellSort(array1D);
    cout << "Sorted 1D array: " << endl;
    print1D(array1D);
    for (int i = 0; i < height; i++)
    {
        delete[] array2D[i];
    }
    delete[] array2D;
    delete[] array1D;
}

void fill2D(float **array2D)
{
    srand(time(NULL));
    for (int i = 0; i < height; i++)
    {
        for (int j = 0; j < width; j++)
        {
            array2D[i][j] = float(rand()) / float((RAND_MAX)) * range;
            array2D[i][j] -= halfRange;
        }
    }
}

void fill1D(float** array2D, float* array1D)
{
    float prod;
    for (int i = 0; i < height; i++)
    {
        prod = 1;
        for (int j = 0; j < width; j++)
        {
            if (array2D[i][j] < 0)
            {
                prod *= array2D[i][j];
            }
        }
        array1D[i] = prod;
    }
}

void shellSort(float* array1D)
{
    float temp;
    int d = height / 2;
    int j;
    while (d >= 1)
    {
        for (int i = d; i < height; i++)
        {
            j = i;
            while (j >= d && array1D[j - d] < array1D[j])
            {
                temp = array1D[j];
                array1D[j] = array1D[j - d];
```

```
            array1D[j - d] = temp;
            j -= d;
        }
      }
      d /= 2;
   }
}



void print1D(float* array1D)
{
   for (int i = 0; i < height; i++)
   {
      printf("%.2f  ", array1D[i]);
   }
   cout << endl << endl;
}

void print2D(float** array2D)
{
   for (int i = 0; i < height; i++)
   {
      for (int j = 0; j < width; j++)
      {
         printf("%7.2f", array2D[i][j]);
      }
      cout << endl;
   }
   cout << endl;
}
```

6. Випробування коду

Результати випробування:



7. Висновки

Було досліджено алгоритм пошуку у двовимірному масиві та сортування в одновимірному, набуто практичних навичок використання цих алгоритмів під час складання програмних специфікацій.