МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Інститут **ІКНІ**

Кафедра **ПЗ**

ЗВІТ

До лабораторної роботи №3

На тему: «Метод сортування Шелла»

З дисципліни: «Алгоритми та структури даних»

Лектор : доцент каф.ПЗ

Коротєєва Т.О.

Виконала: ст.гр.ПЗ-23

Кохман О.В.

Прийняв: асистент каф.ПЗ

Франко А.В.

«____»_____2022 р.

∑____ _ _ _ _ _ _ _ _ .

Львів – 2022

Тема: Алгоритм сортування Шелла.

Мета: Вивчити алгоритм сортування Шелла. Здійснити програмну реалізацію алгоритму сортування Шелла. Дослідити швидкодію алгоритму сортування Шелла.

Теоретичні відомості

Сортування Шелла (англійською «Shell Sort») — це алгоритм сортування, що є узагальненням сортування включенням.

Алгоритм базується на двох тезах:

- Сортування включенням ефективне для майже впорядкованих масивів.
- Сортування включенням неефективне, тому що переміщує елемент тільки на одну позицію за раз.

Тому сортування Шелла виконує декілька впорядкувань включенням, кожен раз порівнюючи і переставляючи елементи, що знаходяться на різній відстані один від одного.

Сортування Шелла не є стабільним.

Сортування Шелла названо начесть автора — Дональда Шелла, який опублікував цей алгоритм у 1959 році.

На початку обираються m елементів: $d_1, d_2, \ldots, d_m$, причому $d_1 > d_2 > \ldots > d_m = 1$.

Потім виконується m впорядкувань методом включення, спочатку для елементів, що стоять через d_1, потім для елементів через d_2і так далі до $d_m = 1$.

Значення $d_1 = m/2$.

Ефективність досягається тим, що кожне наступне впорядкування вимагає меншої кількості перестановок, оскільки деякі елементи вже стали на свої місця.

Оскільки $d_m = 1$, то на останньому кроці виконується звичайне впорядкування включенням всього масиву, а отже кінцевий масив гарантовано буде впорядкованим.

Час роботи залежить від вибору значень елементів масиву d. Існує декілька підходів вибору цих значень:

- Якщо d — впорядкований за спаданням набір чисел виду $(2^i - 1) < n, j \hat{I} n$, то час роботи є $O(N^{1.5})$.
- Якщо d — впорядкований за спаданням набір чисел виду $2^i * 3^j < n/2, i, j \hat{I} n$, то час роботи є $O(N \cdot \log^2 N)$.

Покроковий опис роботи алгоритму сортування Шелла:

Алгоритм Sh:

Задано одновимірний масив array, i,j – індекси проходження по масиву , length – довжина масиву , gap – проміжок, який дорівнює length/2, temp – тимчасова змінна для свапу елементів.

Sh1: цикл, у якому з кожною ітерацією зменшується gap в 2 рази , допоки не дорівнює 1. Виконуються кроки Sh2, Sh3.

Sh2: цикл, у якому ініціалізується i = gap , виконується цикл допоки i<length і з кожною ітерацією і збільшується на 1, виконується крок Sh3.

Sh3: цикл, у якому j = i, з кожним кроком j зменшується на gap , допоки j>= gap. Перевіряємо умову чи array[j-gap] > array[j], якщщо там то свапаємо елементи.

Sh4: Вихід.

Індивідуальне завдання

1. Відвідати лекцію, вислухати та зрозуміти пояснення лектора. Прочитати та зрозуміти методичні вказівки, рекомендовані джерела та будь-які інші матеріали, що можуть допомогти при виконанні лабораторної роботи. Відвідати лабораторне заняття, вислухати та зрозуміти рекомендації викладача.

2. Встановити та налаштувати середовище розробки.

3. Написати віконний додаток на мові програмування C або C++. Реалізована програма повинна виконувати наступну послідовність дій:

1) запитуватиме в користувача кількість цілих чотирьохбайтових знакових чисел — елементів масиву, сортування якого буде пізніше здійснено;

2) виділятиме для масиву стільки пам’яті, скільки необхідно для зберігання вказаної кількості елементів, але не більше;

3) ініціалізовуватиме значення елементів масиву за допомогою стандартної послідовності псевдовипадкових чисел;

4) засікатиме час початку сортування масиву з максимально можливою точністю;

5) сортуватиме елементи масиву в неспадному порядку за допомогою алгоритму сортування Шелла;

6) засікатиме час закінчення сортування масиву з максимально можливою точністю;

7) здійснюватиме перевірку упорядкованості масиву;

8) повідомлятиме користувачу результат перевірки упорядкованості масиву та загальний час виконання сортування з максимально можливою точністю;

9) звільнятиме усю виділену раніше пам’ять.

Варіант 11: задано двовимірний масив дійсних чисел. Замінити максимальні елементи кожного рядка на $\sqrt[3]{x}$. Впорядкувати (переставити) рядки масиву в порядку зростання їх перших елементів.

4. Оформити звіт про виконання лабораторної роботи. Звіт повинен бути надрукований з однієї сторони аркушів формату А4 шрифтом 12 кеглю з одинарним інтерліньяжем та скріплений за допомогою степлера. Правильно оформлений звіт обов’язково повинен містити такі складові частини:

5. Захистити звіт про виконання лабораторної роботи. Процедура захисту передбачає демонстрацію роботи програми, перевірку оформлення звіту та відповіді на будь-яку кількість будь-яких запитань викладача, що так чи інакше стосуються теми лабораторної роботи.

Код програми

Назва файлу:MyForm.h

```
#pragma once
#include "Sort.h"
namespace Project3 {

	using namespace System;
```

```
	using namespace System::ComponentModel;
	using namespace System::Collections;
	using namespace System::Windows::Forms;
	using namespace System::Data;
	using namespace System::Drawing;
	public ref class MyForm : public System::Windows::Forms::Form
	{
	public:
		MyForm(void)
		{
			InitializeComponent();
		}
	protected:
		~MyForm()
		{
			if (components)
			{
				delete components;
			}
		}
	private: System::Windows::Forms::TextBox^ textBox1;
	protected:
	private: System::Windows::Forms::TextBox^ textBox2;
	private: System::Windows::Forms::RichTextBox^ richTextBox1;
	private: System::Windows::Forms::RichTextBox^ richTextBox2;
	private: System::Windows::Forms::RichTextBox^ richTextBox3;
	private: System::Windows::Forms::Button^ button1;
	private: System::Windows::Forms::Label^ label1;
	private: System::Windows::Forms::Label^ label2;
	private: System::Windows::Forms::Label^ label3;
	private: System::Windows::Forms::Label^ label4;
	private: System::Windows::Forms::Label^ label5;
	private: System::Windows::Forms::TextBox^ textBox3;
	private: System::Windows::Forms::TextBox^ textBox4;
	private: System::Windows::Forms::TextBox^ textBox5;
	private: System::Windows::Forms::TextBox^ textBox6;
	private: System::Windows::Forms::Label^ label6;
	private: System::Windows::Forms::Label^ label7;
	private: System::Windows::Forms::Label^ label8;
	private: System::Windows::Forms::Label^ label9;
	private:
		System::ComponentModel::Container ^components;
#pragma region Windows Form Designer generated code
		// another code //

		int rows = 0;
		int columns = 0;
#pragma endregion
	private: System::Void button1_Click(System::Object^ sender,
System::EventArgs^ e) {
		rows = System::Convert::ToInt16(textBox1->Text);
		columns = System::Convert::ToInt16(textBox2->Text);
		Sort* sort = new Sort(rows, columns);
		sort->randomNumbers();
		for (int i = 0; i < sort->getRows(); i++) {
			for (int j = 0; j < sort->getColumns(); j++) {
				richTextBox1->Text += sort-
>array[i][j].ToString("#,0.00") + " ";
			}
			richTextBox1->Text += "\n";
		}
		sort->changeMaxes();
		for (int i = 0; i < sort->getRows(); i++) {
```

```
                    for (int j = 0; j < sort->getColumns(); j++) {
                          richTextBox2->Text += sort-
>array[i][j].ToString("#,0.00") + " ";
                    }
                    richTextBox2->Text += "\n";
              }
              richTextBox3 << sort;
              sort->increaseCounter();
              DateTime start = DateTime::Now;
              textBox3->Text = start.ToString("hh.mm.ss.fff tt");
              sort->shellSort(richTextBox3);
              DateTime end = DateTime::Now;
              textBox4->Text = end.ToString("hh:mm:ss.fff tt");
              TimeSpan inverval = end - start;
              textBox5->Text = inverval.Seconds.ToString() + "." +
inverval.Milliseconds.ToString();
              sort->isOrdered();
              textBox6->Text = sort->getChecked().ToString();
       }
};
}
```

Назва файлу: MyForm.cpp

```
#include "MyForm.h"
using namespace Project3;
int main() {
       Application::EnableVisualStyles();
       Application::SetCompatibleTextRenderingDefault(false);
       Application::Run(gcnew MyForm());
       return 0;
}
```

Назва файлу: Sort.h

```
#ifndef SORT_H
#define SORT_H
#pragma once
#include <random>
#include <cmath>
using namespace std;
class Sort {
public:
       double** array;
private:
       int rows;
       int columns;
       bool isChecked;
       int counter = 0;
public:
       Sort();
       Sort(int _rows, int _columns);
       ~Sort();
       void randomNumbers();
       void changeMaxes();
       void shellSort(System::Windows::Forms::RichTextBox^ richTextBox);
       void isOrdered();
       bool getChecked();
       void increaseCounter();
       int getRows();
       int getColumns();
```

```
	friend void operator<<(System::Windows::Forms::RichTextBox^ richTextBox,
Sort* sort);
};
#endif
```

Назва файлу: Sort.cpp

```
#include "Sort.h"
Sort::Sort() : rows(0), columns(0) {
	array = new double*[rows];
	for (int i = 0; i < rows; i++) {
		array[i] = new double[columns];
	}
}
Sort::Sort(int _rows, int _columns) {
	rows = _rows;
	columns = _columns;
	array = new double*[rows];
	for (int i = 0; i < rows; i++) {
		array[i] = new double[columns];
	}
}
Sort::~Sort() {
	for (int i = 0; i < rows; i++) {
		delete[] array[i];
	}
	delete[] array;
}
void Sort::randomNumbers() {
	random_device random_device;
	mt19937 generator(random_device());
	uniform_real_distribution<double> distribution(0, 200);
	for (int i = 0; i < rows; i++) {
		for (int j = 0; j < columns; j++) {
			array[i][j] = distribution(generator);
		}
	}
}
void Sort::changeMaxes() {
	int i = 0;
	double max = 0;
	int j = 0;
	int countJ = 0;
	while (i < rows) {
		max = array[i][j];
		for (j = 0; j < columns; j++) {
			if (array[i][j] > max) {
				max = array[i][j];
				countJ = j;
			}
		}
		array[i][countJ] = pow(array[i][countJ], 1.0 / 9);
		i++;
	}
}
bool Sort::getChecked() {
	return isChecked;
}
void operator<<(System::Windows::Forms::RichTextBox^ richTextBox, Sort* sort) {
	for (int i = 0; i < sort->rows; i++) {
		for (int j = 0; j < sort->columns; j++) {
```

```
                  richTextBox->Text += sort->array[i][j].ToString("#,0.00") + "
";
            }
            richTextBox->Text += "\n";
      }
      richTextBox->Text += System::Convert::ToString("-------step " + sort-
>counter + " \n");
      sort->increaseCounter();
}
void Sort::shellSort(System::Windows::Forms::RichTextBox^ richTextBox) {
      int col = 0;
      double temp = 0;
      for (int gap = rows / 2; gap > 0; gap /= 2) {
            for (int i = gap; i < rows; i++) {
                  int j;
                  for (int j = i; j >= gap; j = j - gap) {
                        if (array[j - gap][col] > array[j][col]) {
                              while (col < columns) {
                                    temp = array[j][col];
                                    array[j][col] = array[j - gap][col];
                                    array[j - gap][col] = temp;
                                    col++;
                              }
                        }
                        col = 0;
                        richTextBox << this;
                  }
            }
      }
}
void Sort::increaseCounter() {
      counter++;
}
int Sort::getRows() {
      return rows;
}
int Sort::getColumns() {
      return columns;
}
void Sort::isOrdered() {
      int col = 0;
      for (int i = 0; i + 1 < rows; i++) {
            if (array[i][col] > array[i + 1][col]) {
                  isChecked = false;
                  return;
            }
      }
      isChecked = true;
}
```

Протокол роботи

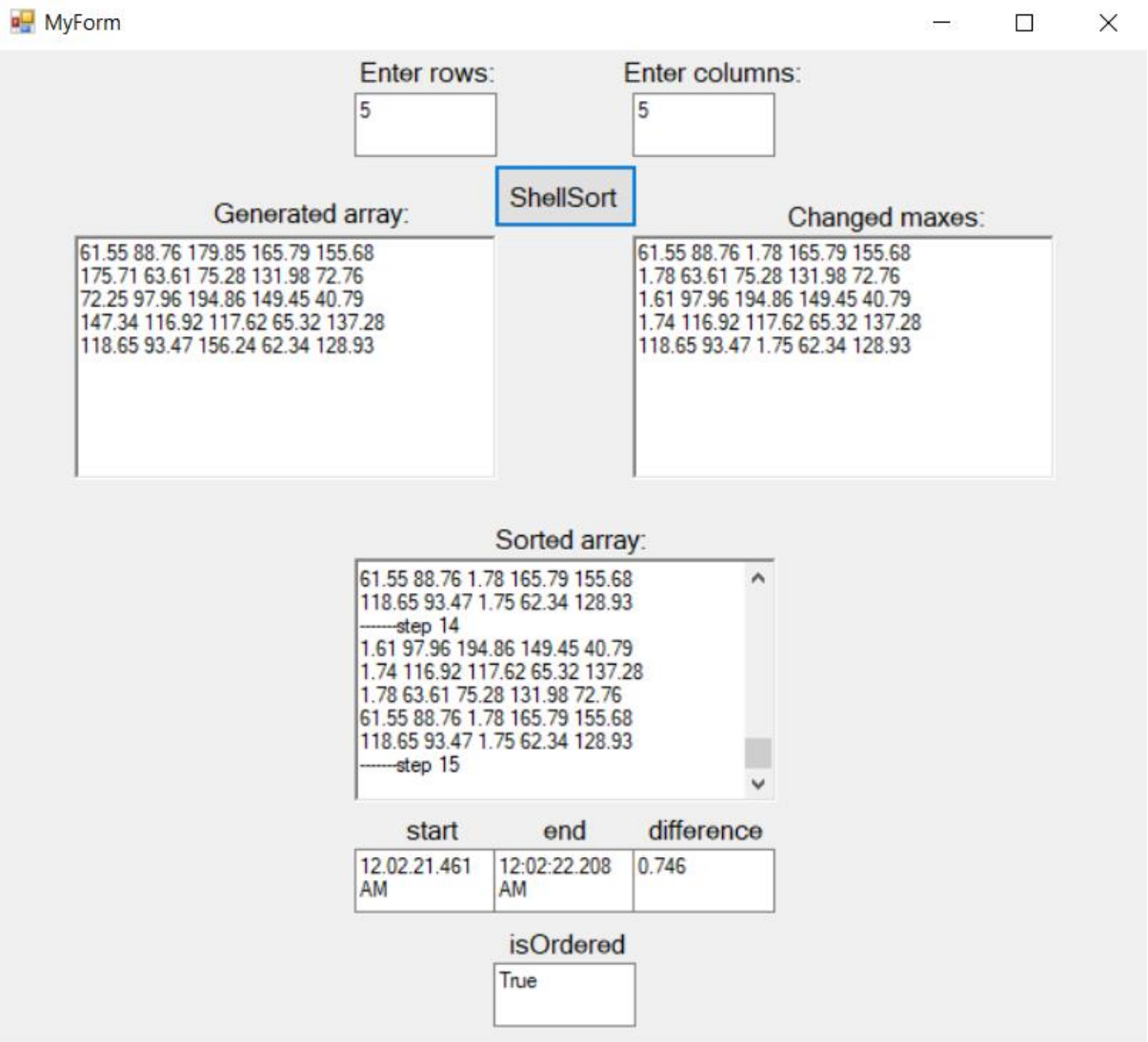

Рис. 1 Форма програми з результатами

Висновок

На цій лабораторній роботі я дізналась про алгоритм сортування Шелла, реалізувала програму, використовуючи цей алгоритм та вивчила алгоритмічну складність алгоритму , що становить в кращому випадку – O(n*log n), в середньому - O(n*log n), та в гіршому – від O(n^1.5) до O(n^2).