МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Інститут **ІКНІ**

Кафедра **ПЗ**

ЗВІТ

До лабораторної роботи №6

На тему: «Метод сортування підрахунком»

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: метод сортування підрахунком.

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

Теоретичні відомості

Сортування підрахунком (англійською «Counting Sort») — алгоритм впорядкування, що застосовується при малій кількості різних елементів (ключів) у масиві даних. Час його роботи лінійно залежить як від загальної кількості елементів у масиві так і від кількості різних елементів.

Ідея алгоритму полягає в наступному: спочатку підрахувати скільки разів кожен елемент (ключ) зустрічається в вихідному масиві. Спираючись на ці дані можна одразу вирахувати на якому місці має стояти кожен елемент, а потім за один прохід поставити всі елементи на свої місця.

В алгоритмі присутні тільки прості цикли довжини N (довжина масиву), та один цикл довжини K (величина діапазону). Отже, обчислювальна складність роботи алгоритму становить $O(N + K)$.

В алгоритмі використовується додатковий масив. Тому алгоритм потребує $E(K)$ додаткової пам'яті.

В такій реалізації алгоритм є стабільним. Саме ця його властивість дозволяє використовувати його як частину інших алгоритмів сортування (наприклад, сортування за розрядами).

Використання даного алгоритму є доцільним тільки у випадку малих K.

Покроковий опис алгоритму сортування підрахунком:

Алгоритм CS:

Задано одновимірний масив цілих чисел array, довжина масиву – length.

CS1: цикл, де визначається максимальний елемент масиву max, створити масив countArray розмірністю max + 1;

CS2: цикл, де i = 0 , виконується за умови , що i<max +1 . У циклі оголошуємо count = 0 , виконуємо крок CS3 і з кожною ітерацією збільшуємо i на 1, countArray[i] = count, count = 0;

CS3: цикл, де j=0, виконується доки j<length і перевіряємо в кожній ітерації чи array[j] == i, якщо так , то count++. Після кожної ітерації збільшуємо j на 1;

CS4: цикл, у якому видозмінюється countArray, i= 1, кожен array[i] = array[i] + array[i-1];

CS5: створюємо масив resultArray розмірністю length, цикл, який виконується за умови i < length, де створюється змінна j = array[i], виконуємо resultArray[countArray[j] – 1] = array[i], countArray[j]--;

CS6: копіюємо значення resultArray в array;

CS7: 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 знак

вих 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 злиттям;

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.

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.

14 варіант:

Задано перелік міст. Упорядкувати за алфавітом лише ті міста, довжина назв яких не перевищує 8.

Код програми

Назва файлу: MyForm.h

```
#pragma once
#include "Sort.h"
namespace Project6 {

	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::Label^ label6;
    protected:
    private: System::Windows::Forms::Label^ label5;
    private: System::Windows::Forms::Label^ label4;
    private: System::Windows::Forms::Label^ label3;
    private: System::Windows::Forms::Label^ label2;
    private: System::Windows::Forms::Label^ label1;
    private: System::Windows::Forms::TextBox^ textBox6;
    private: System::Windows::Forms::TextBox^ textBox5;
    private: System::Windows::Forms::TextBox^ textBox4;
    private: System::Windows::Forms::TextBox^ textBox3;
    private: System::Windows::Forms::RichTextBox^ richTextBox1;
    private: System::Windows::Forms::TextBox^ textBox2;
    private: System::Windows::Forms::TextBox^ textBox1;
    private: System::Windows::Forms::Button^ button1;
    private:
        System::ComponentModel::Container ^components;
#pragma region Windows Form Designer generated code

#pragma endregion
private: System::Void button1_Click(System::Object^ sender, System::EventArgs^ e)
{
    int size = System::Convert::ToInt64(textBox1->Text);
    Sort sort = Sort(size);
    sort.initCities();
    sort.randomNumbers();
    for (int i = 0; i < sort.getSize(); i++) {
        String^ str = gcnew String(sort.getElement(i).c_str());
        textBox2->Text += str + " ";
    }
    DateTime start = DateTime::Now;
    textBox3->Text = start.ToString("hh.mm.ss.fff tt");
    sort.deleteLongCities();
    sort.stringCountingSort(richTextBox1);
    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 Project6;
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 <string>
#include <iostream>
#include <fstream>
using namespace std;
class Sort {
private:
	int length;
	string* array;
	bool isChecked = true;
	string* cities;
	int citiesLength;
	int mainLength = 0;
	string* mainArray;
public:
	Sort();
	Sort(int _length);
	~Sort();
	void randomNumbers();
	void isOrdered();
	bool getChecked();
	void initCities();
	void deleteLongCities();
	string getElement(int index);
	int getSize();
	int convertCharToInt(char value);
	void stringCountingSort(System::Windows::Forms::RichTextBox^ richTextBox);

};
#endif
```

Назва файлу: Sort.cpp

```
#include "Sort.h"
Sort::Sort() {
    length = 0;
    array = new string[length];
    citiesLength = 19;
    cities = new string[citiesLength];
};
Sort::Sort(int _length) {
    length = _length;
    array = new string[length];
    citiesLength = 19;
    cities = new string[citiesLength];
}
Sort::~Sort() {
    delete[] array;
    delete[] cities;
}
void Sort::randomNumbers() {
    random_device random_device;
    mt19937 generator(random_device());
    uniform_int_distribution<> distribution(0, 18);
    int index;
    for (int i = 0; i < length; i++) {
        index = distribution(generator);
        array[i] = cities[index];
    }
}
void Sort::initCities() {
```

```
    string path = "cities.txt";
    ifstream file;
    file.open(path);
    if (!file.is_open()) {
        return;
    }
    int i = 0;
    while (!file.eof() && i < citiesLength) {
        getline(file, cities[i]);
        i++;
    }
    file.close();
}
void Sort::deleteLongCities() {
      for (int i = 0; i < length; i++) {
            if (array[i].length() <= 8) {
                  mainLength++;
            }
      }
      mainArray = new string[mainLength];
      int index = 0;
      for (int i = 0; i < length; i++) {
            if (array[i].length() <= 8) {
            mainArray[index] = array[i];
            index++;
            }
      }
}
string Sort::getElement(int index) {
    return array[index];
}
int Sort::getSize() {
    return length;
}
bool Sort::getChecked() {
    return isChecked;
}
void Sort::isOrdered() {}
void Sort::stringCountingSort(System::Windows::Forms::RichTextBox^ richTextBox) {
      int countArraySize = 26;
      int* countArray = new int[countArraySize] {0};
      for (int i = 0; i < countArraySize; i++) {
            int countI = 0;
            for (int j = 0; j < mainLength; j++) {
                  if (convertCharToInt(mainArray[j].at(0)) == i) {
                        countI++;
                  }
            }
            countArray[i] = countI;
      }
      int* indexArray = new int[countArraySize] {0};
      indexArray[0] = countArray[0];
      for (int i = 1; i < countArraySize; i++) {
            indexArray[i] = indexArray[i - 1] + countArray[i];
      }
      string* resultArray = new string[mainLength]{};
      for (int i = 0; i < mainLength; i++) {
            resultArray[i] = "0";
      }
      int step = 0;
      for (int i = 0; i < mainLength; i++) {
            int j = convertCharToInt(mainArray[i].at(0));
            resultArray[indexArray[j] - 1] = mainArray[i];
```

```
			System::String^ element = gcnew
System::String(mainArray[i].c_str());
			richTextBox->Text += "Step " + step + ": adding " + element + " on
position " + (indexArray[j] - 1) + "\n";
			step++;
			for (int k = 0; k < mainLength; k++) {
				System::String^ str = gcnew
System::String(resultArray[k].c_str());
				richTextBox->Text += str + " ";
			}
			richTextBox->Text += "\n";
			indexArray[j]--;
		}
		for (int i = 0; i < mainLength; i++) {
			mainArray[i] = resultArray[i];
		}
		richTextBox->Text += "\nFinal sorted array:\n";
		for (int i = 0; i < mainLength; i++) {
			System::String^ str = gcnew System::String(mainArray[i].c_str());
			richTextBox->Text += str + " ";
		}
}
int Sort::convertCharToInt(char value) {
		int size = 26;
		char* symbols = new char[size] { 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h',
'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x',
'y', 'z' };
		for (int i = 0; i < size; i++) {
			if ((symbols[i]) == value || toupper(symbols[i]) == value) {
				return i;
			}
		}
		return -1;
}
```

Протокол роботи

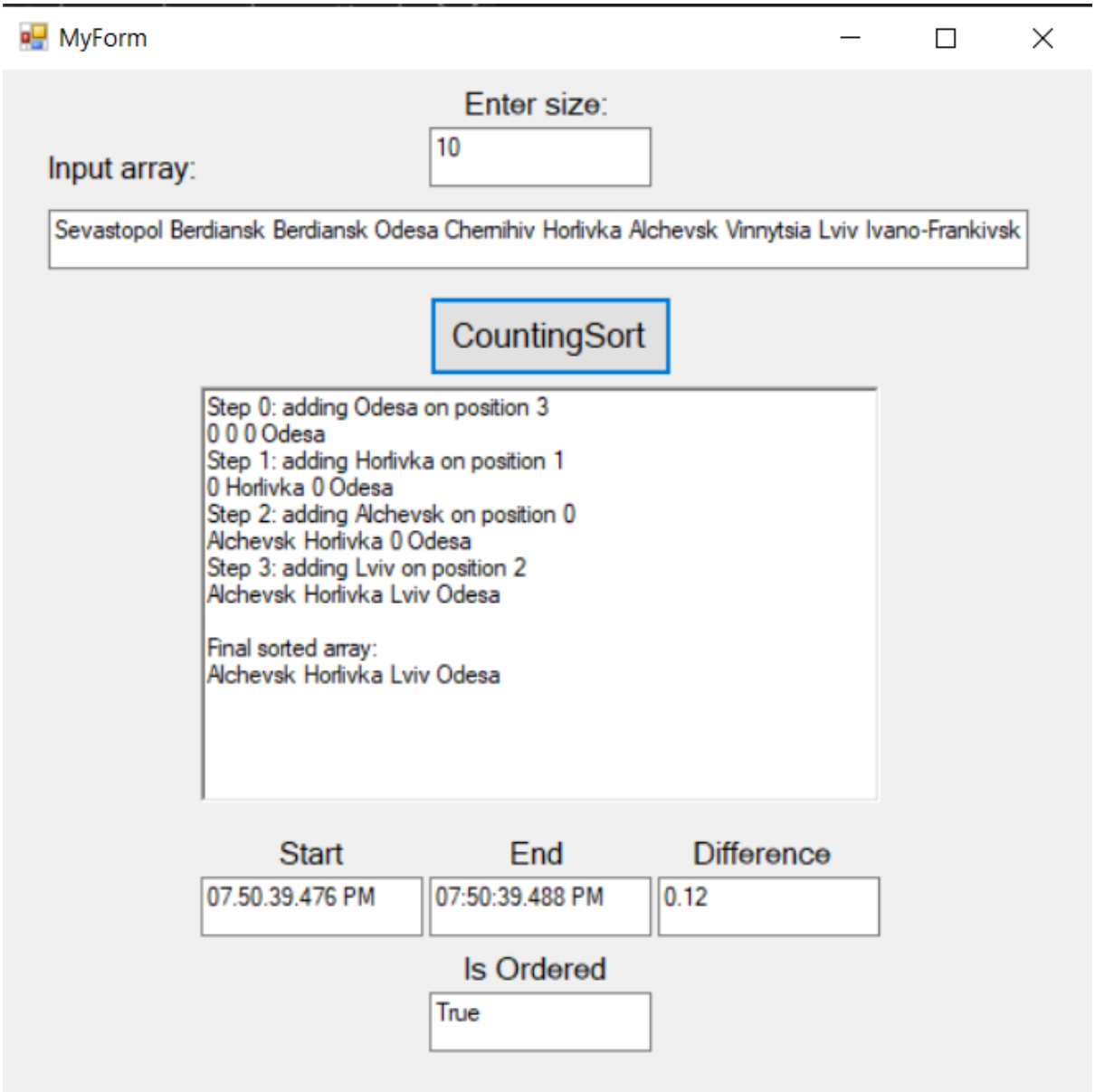

Рис. 1 Результат виконання програми.

Висновок

На цій лабораторній роботі я дізналась про алгоритм сортування підрахунком, реалізувала його на мові c++ та продемонструвала результати роботи програми на формі у Visual Studio 2022. Також дослідила швидкодію алгоритму сортування підрахунком, що дорівнює O(n+k).