**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»**

Інститут **ІКНІ**

Кафедра **ПЗ**

**ЗВІТ**

До лабораторної роботи №2

На тему: «Метод сортування вибором»

З дисципліни: «Алгоритми та структури даних»

**Лектор** : доцент каф.ПЗ

Коротєєва Т.О.

**Виконала:** ст.гр.ПЗ-23

Кохман О.В.

**Прийняв:** аспірант каф.ПЗ

Франко А.В.

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_2022 р.

\_ \_ \_ \_ \_ \_ \_ \_ .

Львів – 2022

**Тема:** Метод сортування вибором.

**Мета:** Вивчити сортування вибором. Здійснити програмну реалізацію алгоритму сортування вибором. Дослідити швидкодію алгоритму сортування вибором.

**Теоретичні відомості**

Сортування вибором (англійською «Selection Sort») — простий алгоритм сортування лінійного масиву, на основі вставок. Велика кількість ітерацій алгоритму робить його неефективним при сортування велеких масивів, і в цілому, менш ефективним за подібний алогоримт сортування включенням. Сортування вибором вирізняється більшою простотою, ніж cортування включенням, і в деяких випадках вищою продуктивністю.

Сортування вибором не є складним в аналізі та порівнянні його з іншими алгоритмами, оскільки жоден з циклів не залежить від даних у списку. Суть алгоритму в тому, що ми вибираємо найменший елемент у невідсортованій частині масиву (від цього і назва – алгоритм “вибору” ), і ставимо його на початок невідсортованої частини масиву. Знаходження найменшого елементу вимагає перегляду усіх n елементів (у даному випадку (n − 1) порівняння), і після цього, перестановки його до першої позиції. Знаходження наступного найменшого елементу вимагає перегляду (n − 1) елементів, і так далі, для (n − 1) + (n − 2) + ... + 2 + 1 = n(n − 1) / 2 порівнянь.

Кожне сканування вимагає однієї перестановки для (n − 1) елементів (останній елемент знаходитиметься на своєму місці).

Покроковий опис алгоритму сортування вибором:

void Sort::selectionSort(System::Windows::Forms::RichTextBox^ richTextBox) {

int min = 0;

int count = 0;

int mainIndex = 0;

int temp = 0;

while (mainIndex < counter) {

min = resultArray[mainIndex];

count = mainIndex;

for (int i = mainIndex; i < counter; i++) {

if (resultArray[i] < min) {

min = resultArray[i];

count = i;

}

}

temp = resultArray[mainIndex];

resultArray[mainIndex] = min;

resultArray[count] = temp;

mainIndex++;

richTextBox << this;

this->increaseCount();

}

}

**Алгоритм S:**

Задано масив resultArray, counter – довжина масиву, min – змінна для збереження мінімального значення масиву, count – змінна для збереження індексу мінімального значення, temp – тимчасова змінна для свапу елементів, mainIndex – змінна для визначення 1 елементу, з якого починаєтьься сортування, i – індекс проходження по масиву.

S1: цикл, який повторюється допоки mainIndex < counter, спочатку присвоює мінімальне значення змінній min у вигляді 1 елементу масиву та індекс count, виконує крок S2 та S3, щоразу збільшує значення mainIndex на 1.

S2: цикл за змінною проходження і, якій присвоюється значення mainIndex, збільшується на 1 після кожного проходження та виконується, доки i<counter. Перевіряється умова чи елемент масиву за індексом і менше мінімального значення, якщо так – присвоюється нове значення min та count.

S3: свап елементу за індексом mainIndex та count за допомогою temp.

S4: 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 С++. 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 вибором;

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 A4 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.

**10)** З двох одновимірних масивів цілих чисел сформувати новий, який включає всі парні числа з першого і непарні з другого масиву. Отриманий масив посортувати в порядку зростання.

**Код програми**

Назва файлу: MyForm.h

#pragma once

#include "Sort.h"

namespace Project2 {

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^ textBox10;

protected:

private: System::Windows::Forms::Label^ label5;

private: System::Windows::Forms::TextBox^ textBox5;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::TextBox^ textBox4;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::TextBox^ textBox3;

private: System::Windows::Forms::TextBox^ textBox2;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::RichTextBox^ richTextBox1;

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::TextBox^ textBox6;

private: System::Windows::Forms::TextBox^ textBox7;

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 size;

#pragma endregion

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e) {

Sort\* sort = new Sort(size);

sort->randomNumbers();

sort->makeResultArray();

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

textBox6->Text += System::Convert::ToString(sort->firstArray[i] + " ");

textBox7->Text += System::Convert::ToString(sort->secondArray[i] + " ");

}

for (int i = 0; i < sort->counter; i++) {

textBox10->Text += System::Convert::ToString(sort->resultArray[i] + " ");

}

richTextBox1 << sort;

sort->increaseCount();

DateTime start = DateTime::Now;

textBox2->Text = start.ToString("hh.mm.ss.fff tt");

sort->selectionSort(richTextBox1);

DateTime end = DateTime::Now;

textBox3->Text = end.ToString("hh:mm:ss.fff tt");

TimeSpan inverval = end - start;

textBox4->Text = inverval.Seconds.ToString() + "." + inverval.Milliseconds.ToString();

sort->isOrdered();

textBox5->Text = sort->getIsChecked().ToString();

}

private: System::Void textBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

size = System::Convert::ToInt16(textBox1->Text);

}

};

Назва файлу: Sort.h

#ifndef SORT\_H

#define \_SORT\_H

#pragma once

#include <random>

using namespace std;

class Sort {

public:

int\* firstArray;

int\* secondArray;

int\* resultArray;

int counter = 0;

private:

int length;

int anotherCounter = 0;

bool isChecked;

public:

Sort();

Sort(int \_length);

~Sort();

void makeResultArray();

void randomNumbers();

void selectionSort(System::Windows::Forms::RichTextBox^ richTextBox);

void isOrdered();

bool getIsChecked();

void increaseCount();

friend void operator<<(System::Windows::Forms::TextBox^ textBox, const Sort\* sort);

friend void operator<<(System::Windows::Forms::RichTextBox^ richTextBox, const Sort\* sort);

};

#endif

Назва файлу: Sort.cpp

#include "Sort.h"

Sort::Sort() : length(0) {

firstArray = new int[length];

secondArray = new int[length];

}

Sort::Sort(int \_length) {

length = \_length;

firstArray = new int[length];

secondArray = new int[length];

}

Sort::~Sort() {

delete[] firstArray;

delete[] secondArray;

}

void Sort::randomNumbers() {

random\_device random\_device;

mt19937 generator(random\_device());

uniform\_int\_distribution<> distribution(-200, 200);

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

firstArray[i] = distribution(generator);

secondArray[i] = distribution(generator);

}

}

bool Sort::getIsChecked() {

return isChecked;

}

void Sort::makeResultArray() {

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

if (firstArray[i] % 2 == 0) {

counter++;

}

}

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

if (secondArray[i] % 2 != 0) {

counter++;

}

}

resultArray = new int[counter];

int tempCounter = 0;

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

if (firstArray[i] % 2 == 0) {

resultArray[tempCounter] = firstArray[i];

tempCounter++;

}

}

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

if (secondArray[i] % 2 != 0) {

resultArray[tempCounter] = secondArray[i];

tempCounter++;

}

}

}

void Sort::isOrdered() {

for (int i = 0; i + 1 < counter; i++) {

if (resultArray[i] > resultArray[i + 1]) {

isChecked = false;

return;

}

}

isChecked = true;

}

void Sort::selectionSort(System::Windows::Forms::RichTextBox^ richTextBox) {

int min = 0;

int count = 0;

int mainIndex = 0;

int temp = 0;

while (mainIndex < counter) {

min = resultArray[mainIndex];

count = mainIndex;

for (int i = mainIndex; i < counter; i++) {

if (resultArray[i] < min) {

min = resultArray[i];

count = i;

}

}

temp = resultArray[mainIndex];

resultArray[mainIndex] = min;

resultArray[count] = temp;

mainIndex++;

richTextBox << this;

this->increaseCount();

}

}

void operator<<(System::Windows::Forms::RichTextBox^ richTextBox, const Sort\* sort) {

for (int i = 0; i < sort->counter; i++) {

richTextBox->Text += sort->resultArray[i].ToString() + " ";

}

richTextBox->Text += System::Convert::ToString("-------step " + sort->anotherCounter + " \n");

}

void Sort::increaseCount() {

anotherCounter++;

}

**Протокол роботи**

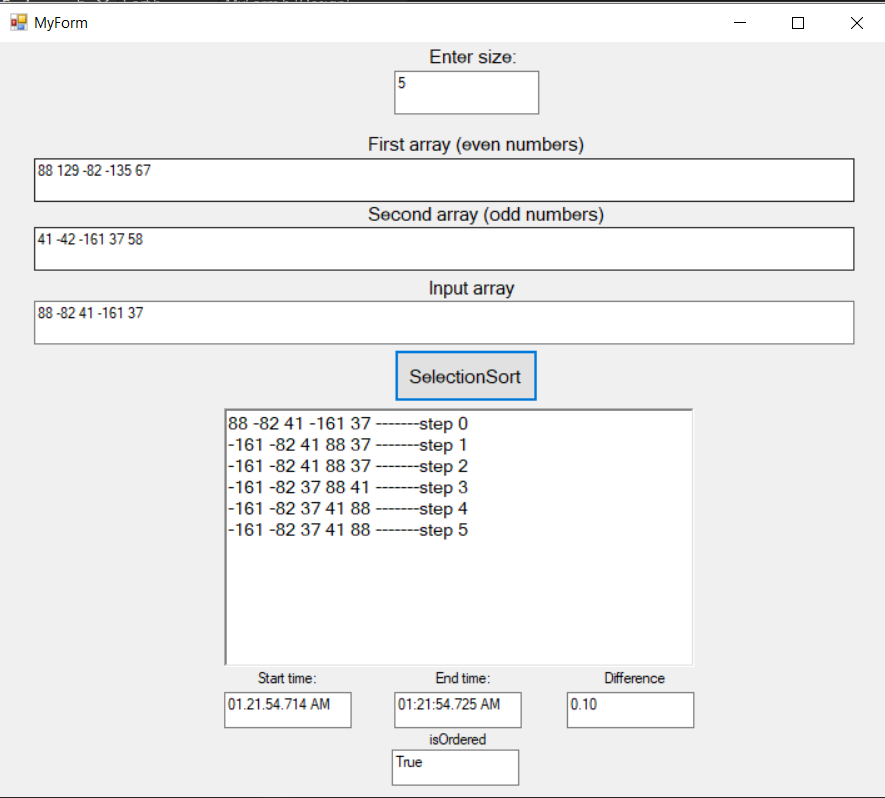
****

Рис. 1 Форма програми з результатами сортування

**Висновок**

На даній лабораторній роботі я дізналась про метод сортування вибором та реалізувала віконний проект у Visual Studio 2022 з використанням цього алгоритму сортування. Швидкоція цього алгориму в найгіршому випадку – O(n^2), у середньому – O(n^2) та у найкращому – O(n^2), що свідчить про тте, що алгоритм є неефективним по затратам ресурсів при великій кількості даних.