**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра «Информационные системы и технологии»**

отчет

**по лабораторной работе №2**

**по дисциплине «основы алгоритмизации и программирования на языке C++»**

Тема: одномерные статические массивы

|  |  |  |
| --- | --- | --- |
| Студент гр. 0323 |  | Ткаченко Ю.В. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2020

**Цель работы.**

Изучение структурной организации массивов и способов доступа к их элементам, совершенствование навыков процедурного при решении задач обработки статических одномерных массивов, практическая реализация разных методов сортировки и анализ их эффективности.

**Основные теоретические положения.**

**Постановка задачи.**

Необходимо написать программу, которая:

1. Создает целочисленный массив размерности N = 100. Элементы массивы должны принимать случайное значение в диапазоне от -99 до 99.
2. Отсортировать заданный в пункте 1 элементы массива […] сортировкой (от меньшего к большему). Определить время, затраченное на сортировку, используя библиотеку chrono.
3. Найти максимальный и минимальный элемент массива. Подсчитайте время поиска этих элементов в отсортированном массиве и неотсортированном, используя библиотеку chrono.
4. Выводит среднее значение (если необходимо, число нужно округлить) максимального и минимального значения. Выводит индексы всех элементов, которые равны этому значению, и их количество.
5. Выводит количество элементов в отсортированном массиве, которые меньше числа a, которое инициализируется пользователем.
6. Выводит количество элементов в отсортированном массиве, которые больше числа b, которое инициализируется пользователем.
7. Выводит информацию о том, есть ли введенное пользователем число в отсортированном массиве. Реализуйте алгоритм бинарного поиска. Сравните скорость его работы с обычным перебором. (\*)
8. Меняет местами элементы массива, индексы которых вводит пользователь. Выведите скорость обмена, используя библиотеку chrono.

Должна присутствовать возможность запуска каждого пункта многократно.

Варианты сортировок:

|  |  |
| --- | --- |
| № | Название сортировки |
| 1 | Bubble sort (пузырьковая сортировка) |
| 2 | Shaker sort (шейкер-сортировка) |
| 3 | Comb sort (сортировка расчёской) |
| 4 | Insert sort (сортировка вставками) |
| 5 (\*) | Quick sort (быстрая сортировка) |

Сделать вывод по проделанной работе.

**Выполнение работы.**

Для решения поставленной задачи была написана программа в процедурной парадигме на языке C++. Итоговый код программы представлен в приложении А.

**Выводы.**

Приложение А

Полный код программы

#include <iostream>

#include <ctime>

#include <chrono>

#include <algorithm>

#include <numeric>

//#include <iomanip>

using namespace std;

using namespace chrono;

void bubbleSort(int a[])

{

for (int i = 0; i < 10; i++)

{

for (int j = 0; j < 10 - i - 1; j++)

{

if (a[j] > a[j + 1])

{

//swap(a[j] , a[j+1]);

int temp = a[j];

a[j] = a[j + 1];

a[j + 1] = temp;

}

}

}

}

void sort()

{

const int SIZE = 10;

int N[SIZE], min, max, midle;

srand(time(NULL)); // инициализация генерации случайных чисел

cout << endl;

for (int i = 0; i < 10; i++)

{

N[i] = (rand() % 100 - 35);

}

cout << "Before Sorting: " << " ";

for (int i = 0; i < 10; i++)

{

cout << N[i] << " ";

}

cout << endl;

cout << endl;

bubbleSort(N);

cout << "After Sorting: " << " ";

for (int i = 0; i < 10; i++)

{

cout << N[i] << " ";

}

system\_clock::time\_point start = system\_clock::now();

cout << "\n";

system\_clock::time\_point end = system\_clock::now();

duration<double> sec = end - start;

cout << endl;

cout << "seconds: " << sec.count() << endl;

/\*

cout << "Task 3" << endl;

N[SIZE];

int size1 = sizeof(N[SIZE] / sizeof(int));

int summ = accumulate(&N[0], &N[size1], 0);

int mean = summ / size1;

cout << mean << endl;

for (int i = 0; i < size1; i++)

{

cout << N[i] << " ";

}

\*/

/\*

int a1[5];

for (int i = 0; i < 5; i++)

{

if (a1[i] == N[i]);

}

cout << endl;

int b1[5];

for (int i = 0; i < 5; i++)

{

if (b1[i] == N[i+1])

}

\*/

cout << endl;

cout << endl;

int a;

cout << "Insert the number: " << " ";

cin >> a;

cout << "less than a : " << " ";

for (int i = 0; i < 10; ++i)

{

if (a < N[i]) { cout << N[i] << " "; }

}

cout << endl;

cout << endl;

int b;

cout << "Insert the number: " << " ";

cin >> b;

cout << "number more a : " << " ";

for (int i = 0; i < 10; ++i)

{

if (b > N[i]) { cout << N[i] << " "; }

}

cout << endl;

cout << endl;

cout << endl;

cout << "-----------------------------------------------------" << endl;

min = N[0];

max = N[0];

for (int i = 1; i < SIZE; i++)

{

if (N[i] < min) { min = N[i]; }

if (N[i] > max) { max = N[i]; }

}

midle = N[0];

for (int i = 1; i < SIZE; i++)

{

if (N[i] = max / min) { midle = N[i]; }

}

cout << "Min: " << min << endl;

cout << "Max: " << max << endl;

cout << "Midle: " << midle << endl;

cout << endl;

//cout << "Number a: " << a << endl;

cout << endl;

}

int main()

{

sort();

system("pause");

}

Приложение Б

результаты тестирования

