**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационных систем (ИС)**

отчет

**по лабораторной работе №2**

**по дисциплине «Программирование»**

Тема: Одномерные статические массивы

|  |  |  |
| --- | --- | --- |
| Студент гр. 1323 |  | Съедин А.С. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2021

**Цель работы.**

Изучить структуру одномерных массивов и обработку их данных; изучить различные виды сортировок; определить время работы действий с массивами.

**Основные теоретические положения.**

Массив представляет собой индексированную последовательность однотипных элементов с заранее определенным количеством элементов.

Все массивы можно разделить на две группы: одномерные и многомерные. Описание массива в программе отличается от объявления обычной переменной наличием размерности массива, которая задается в квадратных скобках после имени.

Элементы массива нумеруются с нуля. При описании массива используются те же модификаторы (класс памяти, const и инициализатор), что и для простых переменных.

Сортировка – процесс размещения элементов заданного множества объектов в определенном порядке. Когда элементы отсортированы, их проще

найти, производить с ними различные операции. Сортировка напрямую влияет

на скорость алгоритма, в котором нужно обратиться к определенному элементу массива.

Простейшая из сортировок – сортировка обменом (bubble sort). Вся суть метода заключается в попарном сравнении элементов и последующем обмене. Таким образом, если следующий элемент меньше текущего, то они меняются местами, максимальный элемент массива постепенно смещается в конец массива, а минимальный – в начало. Один полный проход по массиву может гарантировать, что в конце массива находится максимальный элемент. Затем процесс повторяется до тех пор, пока вся последовательность не будет упорядочена.

Одна из самых быстрых сортировок – Quick sort. Эта сортировка, по сути,

является существенно улучшенной версией алгоритма пузырьковой сортировки.

Общая идея алгоритма состоит в том, что сначала из массива выбирается

элемент, который называется опорным. От выбора опорного элемента не зависит корректность алгоритма, но в отдельных случаях может сильно зависеть его эффективность. Затем необходимо сравнить все остальные элементы с опорным и переставить их в массиве так, чтобы разбить его на три непрерывных отрезка, следующие друг за другом: меньше опорного, равный опорному и больше опорного. Для меньших и больших значений необходимо выполнить рекурсивно ту же последовательность операций, если длина отрезка больше единицы.

Алгоритм бинарного поиска – классический алгоритм поиска в отсортированном массиве, который использует дробление массива на половины. Если элемент, который необходимо найти, присутствует в списке, то бинарный поиск возвращает ту позицию, в которой он был найден.

Сам алгоритм имеет следующий вид:

1. Определение значения в середине массива (или иной структуры данных). Полученное значение сравнивается с ключом (значением, которое необходимо найти).

2. Если ключ меньше значения середины, то необходимо осуществлять

поиск в первой половине элементов, иначе – во второй.

3. Поиск сводится к тому, что вновь определяется значение серединного

элемента в выбранной половине и сравнивается с ключом.

4. Процесс продолжается до тех пор, пока не будет определен элемент,

равный значению ключа, или не станет пустым интервал для поиска

**Ход работы**

1. Создан целочисленный массив размерности N=100 со случайными числами от -99 до 99;
2. Реализовано меню выбора сортировок для пользователя;
3. В п.2 выполнена сортировка bubble sort. Также с помощью библиотеки chrono выполнен подсчёт времени, затраченного на выполнение данной сортировки;
4. В п.2 выполнена сортировка insert sort. Также с помощью библиотеки chrono выполнен подсчёт времени, затраченного на выполнение данной сортировки;
5. Выведен максимальный и минимальный элемент в неотсортированном и отсортированном массиве;
6. Выведено среднее значение максимального и минимального элемента, также индексы элементов, равному этому значению, и их количество;
7. Выведены количество элементов, которые меньше/больше чисел a и b, инициализируемых пользователем;
8. Поменяны местами элементы массива, индексы которых вводит пользователь, выведена скорость обмена с помощью библиотеки chrono.

**Результат работы программы**

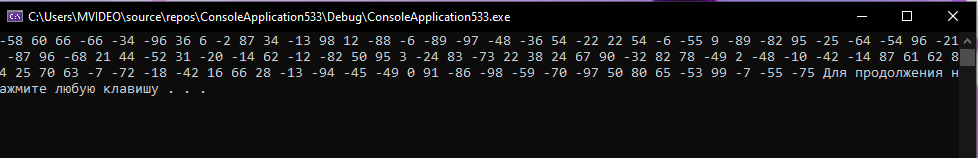


Рисунок 1

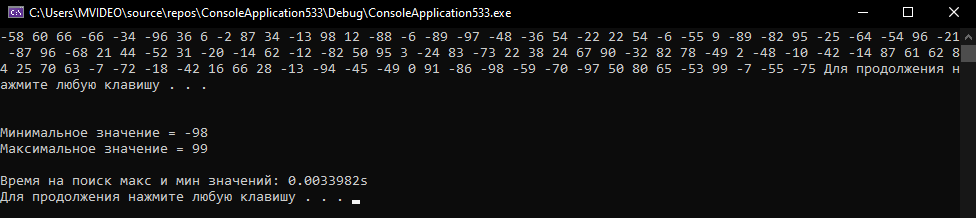


Рисунок 2

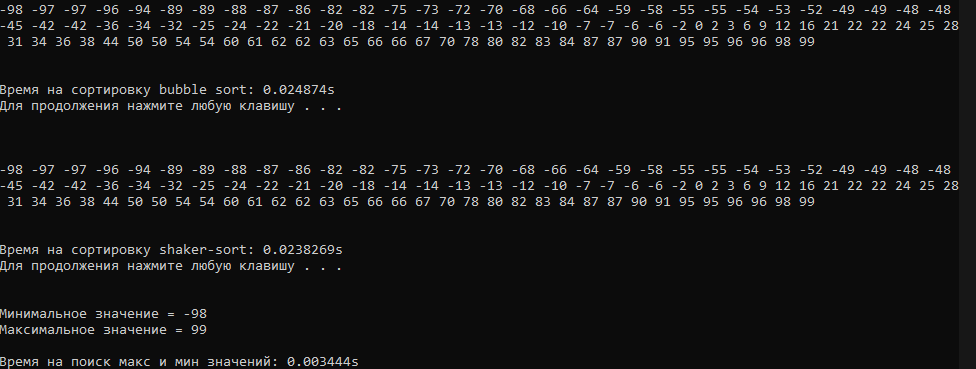


Рисунок 3

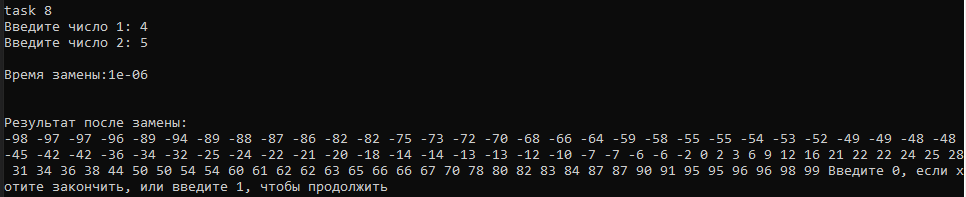
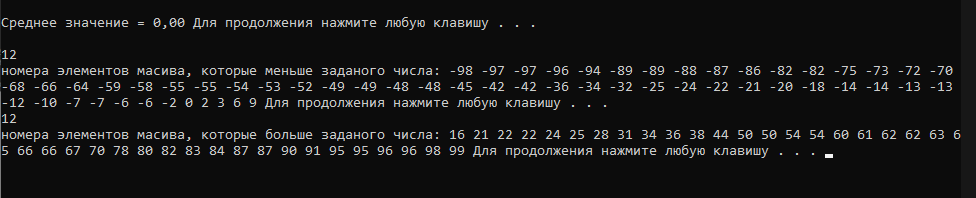


Рисунок 4

**Вывод.**

Изучена структура одномерных массивов и обработка их данных; изучены и выполнены различные виды сортировок; определено время работы действий с массивами.