**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационных систем**

отчет

**по лабораторной работе №2**

**по дисциплине «Программирование»**

Тема: Одномерные статические массивы

|  |  |  |
| --- | --- | --- |
| Студент гр. 1323 |  | Иванов Р.И. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2021

**Цель работы.**

Ознакомиться со структурой одномерного массива, обработкой данных одномерных массивов; изучить различные виды сортировок; провести временную оценку различных 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. Размерность массива вместе с типом его элементов определяет объем памяти, необходимый для размещения массива, которое выполняется на этапе компиляции, поэтому размерность должна быть задана целой положительной константой или константным выражением.

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.

**Пузырьковая сортировка массива (bubble sort)**

Простейшая из сортировок – сортировка обменом (пузырьковая сортировка). 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.

**Шейкер-сортировка массива (shaker sort)**

Shaker sort – модификация пузырьковой сортировки. Принцип работы этой сортировки аналогичен bubble sort: попарное сравнение элементов и последующий обмен местами. Но имеется существенное отличие. Как только максимальный элемент становится на свое место, алгоритм не начинает новую итерацию с первого элемента, а запускает сортировку в обратную сторону. Алгоритм гарантирует, что после выполнения первой итерации, минимальный и максимальный элемент будут в начале и конце массива соответственно.

Затем процесс повторяется до тех пор, пока массив не будет отсортирован. За счет того, что сортировка работает в обе стороны, массив сортируется на порядок быстрее.

**Сортировка массива расчёской (comb sort)**

Очевидный недостаток bubble и shaker sort заключается в том, что элементы переставляются максимум на одну позицию.

Comb sort (сортировка расческой) – ещё одна модификация сортировки пузырьком. Алгоритм был разработан специально для случаев, когда минимальные элементы стоят слишком далеко, или максимальные – слишком близко к началу массива. В сортировке расческой переставляются элементы, стоящие на расстоянии.

Оптимально изначально взять расстояние равным длине массива , а далее уменьшать его на определенный коэффициент, который примерно равен 1.247. Когда расстояние станет равно 1, выполняется обычная сортировка пузырьком.

**Сортировка массива вставками (insert sort)**

Сортировка вставками (insert sort) – алгоритм сортировки, в котором элементы массива просматриваются по одному, и каждый новый элемент размещается в подходящее место среди ранее упорядоченных элементов.

Общая суть сортировки вставками такова:

1) Перебираются элементы в неотсортированной части массива.

2) Каждый элемент вставляется в отсортированную часть массива на то место, где он должен находится.

Сортировка вставками делит массив на 2 части – отсортированную и неотсортированную. С каждым новым элементом отсортированная часть будет увеличиваться, а неотсортированная уменьшаться. Причем найти нужное место для очередного элемента в отсортированном массиве достаточно легко.

**Быстрая сортировка массива (quick sort)**

Быстрая сортировка (quick sort) – одна из самых быстрых сортировок. Эта сортировка по сути является существенно улучшенной версией алгоритма пузырьковой сортировки.

Общая идея алгоритма состоит в том, что сначала выбирается из массива 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 массив на три непрерывных отрезка, следующие друг за другом: меньше опорного, раны опорному и больше опорного. Для меньших и больших значений необходимо выполнить рекурсивно ту же последовательность операций, если длина отрезка больше единицы.

Бинарный поиск

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

**Работа программы:**

1. Реализована возможность вызова других функций многократно;
2. В пределах функции «Создание\_Массива» заполняет целочисленный массив размерности N = 100 случайными значениями в диапазоне от –99 до 99;
3. В пределах функции «Пузырьковая\_сортировка» сортирует заданные в задании 1 элементы массива. Определяя время, затраченное на сортировку. Используется библиотека chrono;
4. В пределах функции «Найти\_максимальный\_и\_минимальный\_элемент\_массива» находит максимальный и минимальный элементы массива. Определяется время поиска этих элементов в отсортированном массиве и неотсортированном (который в процессе создаётся заново). Используется библиотека chrono;
5. В пределах функции «Вывести\_среднее\_значение» выводит среднее значение (округляя число) максимального и минимального значения и выводит все числа, которые равны этому значению;
6. В пределах функции «Количество\_элементов\_меньше\_числа\_a» выводит количество элементов в отсортированном массиве, которые меньше числа a (вводится пользователем);
7. В пределах функции «Количество\_элементов\_больше\_числа\_b» выводит количество элементов в отсортированном массиве, которые больше числа b (вводится пользователем);
8. В пределах функции «Бинарный\_поиск» выводит информацию о том, есть ли введенное пользователем число в отсортированном массиве. Реализован алгоритм бинарного поиска. Произведено сравнение скорости его работы со скоростью обычного перебора (линейным поиском);
9. В пределах функции «Поменять\_элементы\_местами» меняет местами элементы массива, индексы которых вводит пользователь. Определяется скорость обмена. Используется библиотека chrono.

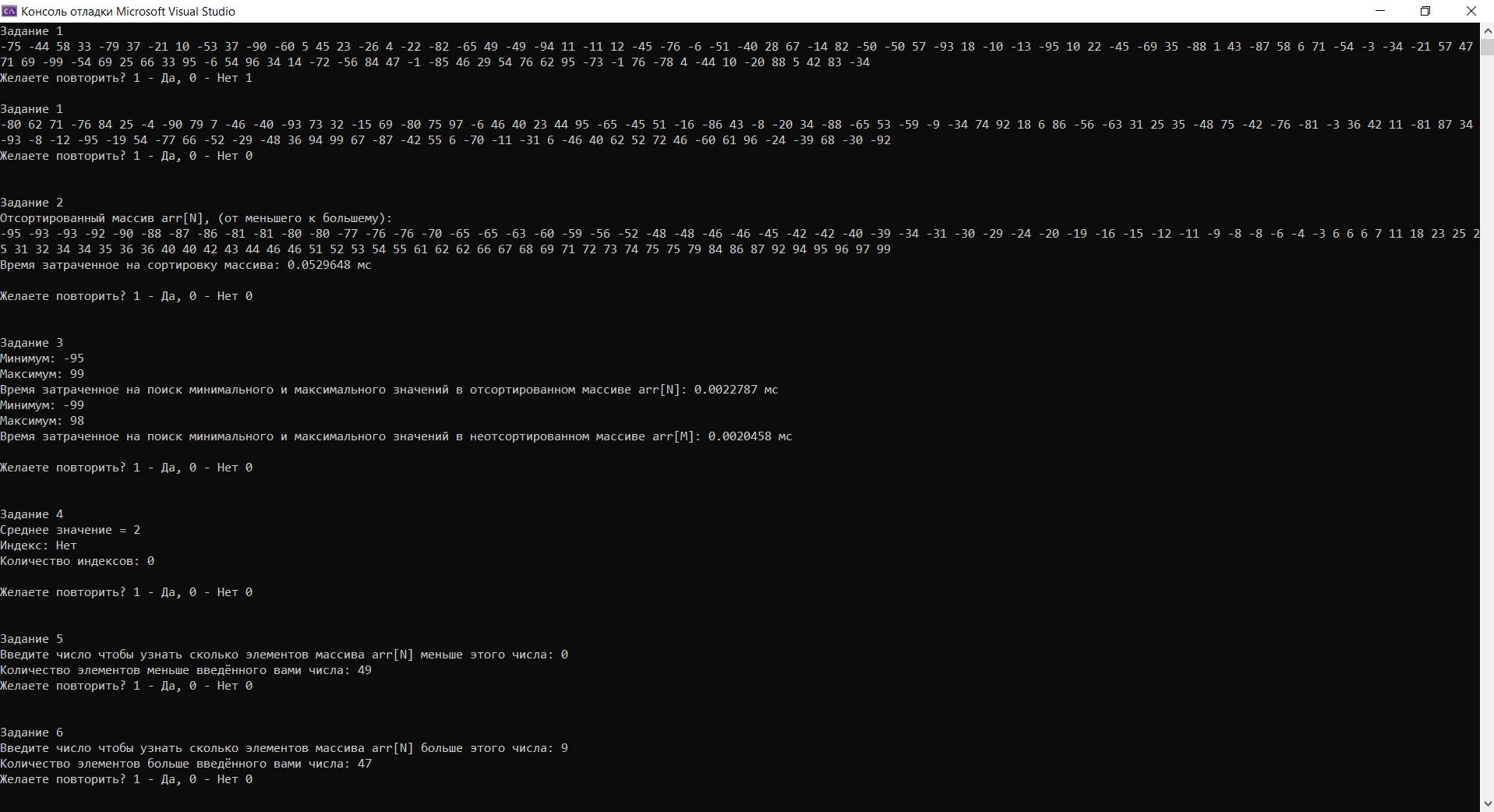
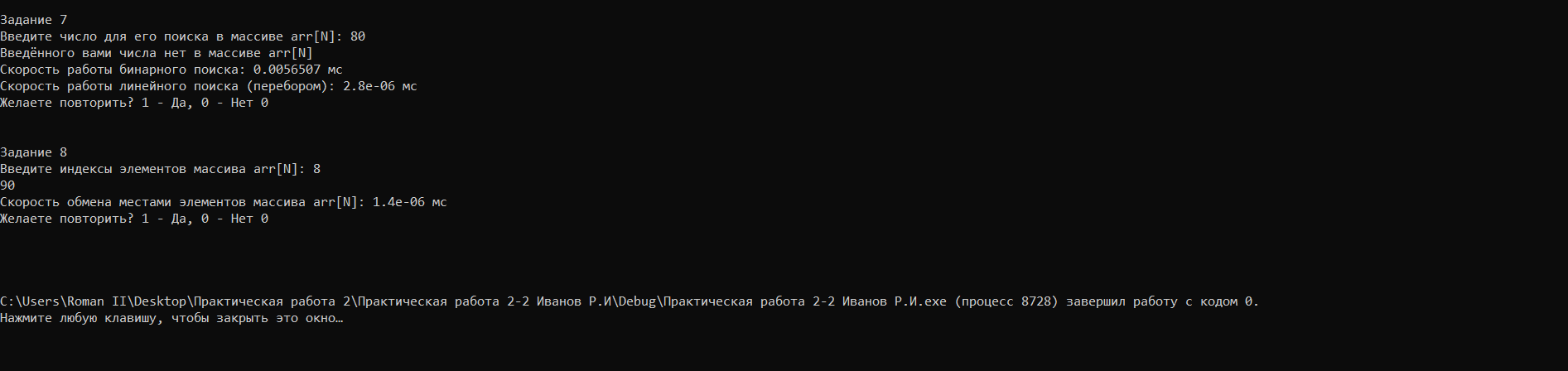
**Результат работы программы**

Рисунок 1. Работа программы с задания 1 по задание 6, с отработкой многократного вызова (задание 1).

Рисунок 2. Работа программы задания 7 и 8. Конец программы.

**Выводы.**

Были осуществлены: работа со структурой одномерного массива, обработка данных одномерных массивов; теоретически изучены различные виды сортировок, а пузырьковая сортировка на практике; реализована сортировка Bubble sort; Измерено время работы различных действий с массивами.