МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №4
по дисциплине «Алгоритмы и структуры данных»
Тема: Алгоритмы сортировки

Студент гр. 9303 _____ Ахримов А.М.

Преподаватель _____ Филатов А.Ю.

Санкт-Петербург
2020

Цель работы.

Реализовать двухпутевую сортировку бинарными вставками.

Задание.

3 вариант. Двухпутевая сортировка бинарными вставками (при каждой вставке перемещаются не более половины элементов отсортированной последовательности).

Основные теоретические положения.

Двухпутевая сортировка бинарными вставками это более оптимизированная версия сортировки простыми вставками.

Для начала рассмотрим метод простых вставок. Задача заключается в следующем: есть часть массива, которая уже отсортирована, и требуется вставить остальные элементы массива в отсортированную часть, сохранив при этом упорядоченность. Для этого на каждом шаге алгоритма мы выбираем один из элементов входных данных и вставляем его на нужную позицию в уже отсортированной части массива, до тех пор пока весь набор входных данных не будет отсортирован. Метод выбора очередного элемента из исходного массива произволен.

Вместо линейного поиска позиции можно использовать бинарный поиск, что уменьшит количество сравнений с $O(N^2)$ до $O(N \log N)$. Количество сравнений заметно уменьшилось, но для того, чтобы поставить элемент на своё место, всё ещё необходимо переместить большое количество элементов. В итоге время выполнения алгоритма в асимптотически не уменьшилось.

Но и это можно оптимизировать путём двухпутевых вставок. Суть этого метода в том, что вместо отсортированной части массива мы используем область вывода. Первый элемент помещается в середину области

вывода, а место для последующих элементов освобождается путём сдвига элементов влево или вправо туда, куда выгоднее. Благодаря тому что для вставки j-ого элемента потребуется j/2 сдвигов в худшем случае вместо j, то и итоговое число необходимых операций в худшем случае составит $N^2/4+NlogN$.

Выполнение работы.

Программа считывает данные с файла и выводит результат в терминал.

Шаблонная функция `binSearch()` ищет в данном массиве место для нового элемента. Функция принимает в качестве аргументов: массив данных, элемент, который требуется вставить в массив, левую и правую границы поиска в массиве. Функция основана на бинарном поиске. Она берёт средний элемент (индекс считается по границам поиска в массиве) и сравнивает его с новым, если новый больше - смещает левую границу на индекс среднего элемента, меньше — смещает правую границу. Так продолжается до тех пор пока разница между правой и левой границей не станет меньше единицы.

Основная функция — `sort()`, она принимает массив данных и его размер. Она создаёт новой массив того же размера, что и исходный массив. Первое значение из исходного массива ставится в середину нового. Устанавливаются границы отсортированного массива. Далее функция проходится по всем элементам исходного массива и запускает для каждого элемента бинарный поиск. Если найденный индекс ближе к левой границе массива, сдвигает левую часть массива, если к правой — правую. С каждым вставкой нового элемента расширяются границы отсортированного массива.

В теории данная сортировка должна хорошо работать для вставки небольшого количества элементов в новый массив. Но из-за того, что приходится постоянно смещать часть массива, данная сортировка не является оптимальной для больших наборов данных.

На практике ситуация осложняется тем, что нужно постоянно следить за границами отсортированного массива, что значительно осложняет его для реализации. Также может возникнуть проблема при сдвигах массива: если верхняя граница отсортированного массива будет упираться в размер исходного массива, то придётся двигать левую часть массива, даже если это не оптимально. При неудачном стечении обстоятельств данный алгоритм по количеству операций может превышать теоритический худший вариант $N^2/4+NlogN$.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные
1.	5 5 7 3 4 6	5 5 7 3 5 7 3 4 5 7 3 4 5 6 7 Sorted array: 3 4 5 6 7
2.	12 10 3 2 6 9 8 122 1 80 10 12 13	10 3 10 2 3 10 2 3 6 10 2 3 6 9 10 2 3 6 8 9 10 2 3 6 8 9 10 122 1 2 3 6 8 9 10 122 1 2 3 6 8 9 10 80 122 1 2 3 6 8 9 10 10 80 122

		1 2 3 6 8 9 10 10 12 80 122 1 2 3 6 8 9 10 10 12 13 80 122 Sorted array: 1 2 3 6 8 9 10 10 12 13 80 122

Выводы.

В ходе выполнения работы был изучен алгоритм двухпутевой сортировки бинарными вставками. Были изучены его достоинства и недостатки. Было установлено, что данную сортировку не рекомендуется использовать для больших массивов.