**Лабораторная работа №2 (Отчет)**

**Марченко Максим 9 группа**

**8 ВАРИАНТ (9-ого не было)**

Разработать алгоритм и программу ускоренного линейного поиска. В качестве исходных данных использовать строку текста, из которой необходимо выделить слова. Затем слова упорядочить по алфавиту. Аргумент поиска – слово.

**1.Название и цель работы.**

ИССЛЕДОВАНИЕ И ОЦЕНКА АЛГОРИТМОВ ПОИСКА

***Цель работы:*** Разработка программ, реализующих различные алгоритмы поиска, и оценка их временной и пространственной сложности.

**2. Словесное описание заданного алгоритма поиска.**

Улученный линейный поиск заключается в том, что мы, когда находим нужное слово, прекращаем его искать. В обычном линейном поиске мы бы гарантированно проходили по всем элементам и асимптотическая оценка сложности алгоритма была бы O(n). А так будет примерно n/2 и 1 в лучшем случае.

**3. Текст программы.**

*start = System.nanoTime();*

*int position = SearchWord(words, word);*

*end = System.nanoTime();*

*private static int SearchWord(String[] str, String word){*

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

*if(str[i].equals(word)) return i;*

*}*

*return -1;*

*}*

**4.Формулы верхней оценки временной и емкостной сложности заданного алгоритма.**

***Верхняя оценка алгоритма:*** O(n), так как в алгоритме присутствует цикл.

***Асимптотическая оценка алгоритма для лучшего случая:*** Лучший случай будет тогда, когда нужное слово стоит на первом месте в массиве, тогда:

Θ(n) = 1(вызов функции SearchWord) + 1(Инициализация i) + 1(присваивание i нуля) + 1(сравнение i с количеством слов) + 2(индексация одномерного массива) + 1(сравнение) + 1(возвращение индекса) = 8 элементарных операций

***Асимптотическая оценка алгоритма в худшем случае:*** Худший случай будет тогда, когда слово в массиве стоит в конце или вовсе отсутствует, тогда:

Θ(n) = 1(вызов функции SearchWord) + 1(инициализация i) + 3n(сюда входит присваивание i нового значения, проверка условия и увеличение i на 1, где n количество слов) + n\*(2(индексация одномерного массива) + 1 (сравнение)) + 1 (возвращение индекса) =

6n + 3 элементарных операций

**5.Результаты экспериментальной оценки временной и емкостной сложности заданного алгоритма.**

*Время выполнения поиска в лучшем случае: 2.2456E-5 секунд!*

*Время выполнения поиска в худшем случае: 2.8071E-5 секунд! для 39 слов*