Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
По практической работе №3
Дисциплины «Алгоритмизация»

Выполнил:

Пустяков Андрей Сергеевич

2 курс, группа ИВТ-б-о-22-1,

09.03.01 «Информатика и вычислительная техника (профиль) «Программное обеспечение средств вычислительной техники и автоматизированных систем», очная форма обучения

_____

(подпись)

Руководитель практики:

Воронкин Р. А. кандидат технических наук, доцент, доцент кафедры инфокоммуникаций

_____

(подпись)

Ставрополь, 2023 г.

Тема: Алгоритм линейного поиска, метод наименьших квадратов.

Цель: изучить алгоритм линейного поиска, изучить метод наименьших квадратов, рассмотреть скорость нахождения элемента в списке при двух случаях расположения необходимого элемента: средний – когда элемент находится в середине списка, худший – когда нужного элемента нет в списке.

Ход работы:

Необходимо создать программу, которая создает данные для анализа скорости работы линейного поиска в списке в среднем случае (если искомый элемент находится в середине списка) и в худшем случае (если искомый элемент не находится в списке, и программа проверяет каждый элемент списка последовательно).

На основе полученных данных о затраченном времени на поиск элемента в списках различной длины необходимо построить точечный график. С использованием метода наименьших квадратов необходимо построить график функции.

Код программы данного задания на языке программирования Python и результаты работы программы (данные о времени поиска в двух разных текстовых файлах) (рис. 1, 2, 3). Данная программа находит элемент в списке или проходит по нему полностью в худшем случае с различными размерами списков. Полученные данные помещаются в два отдельных файла (для среднего и худшего случая «average_case.txt» и «wotst_case.txt»)

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import timeit  # импорт модуля для измерения времени


def search(list_search, desired_value):
    """
    Функция поиска заданного значения в переданном ей списке списке.
    """
    for i in list_search:
        if desired_value == i:
            break
    return None
```

```
if __name__ == '__main__':
    # Необходимо проанализировать скорость нахождения элемента в зависимости
от его положения в списке
    # методом линейного поиска. Необходимо рассмотреть два случая нахождения
элемента в списке:
    # - средний случай (элемент находится в середине списка);
    # - худший случай (элемент находится в конце списка).
    # Заполним списки известными целыми значениями в заданном диапазоне,
тогда элемент находящийся в
    # середине будет равен половине диапазона, а элемент, которого нет в
списке будет отрицательного значения.

    current_list = [i for i in range(1, 100001)]

    # Для записи результатов используются два файла в той же директории, что
и данный модуль.
    filename_average = 'average_case.txt'
    filename_worst = 'worst_case.txt'

    # Получим данные при поиске элемента в списке в среднем случае.
    for number in range(0, 400):

        # Получим данные при поиске элемента в списке в среднем случае.
        middle_element = len(current_list) / 2
        with open(filename_average, 'a') as file:
            time_search = timeit.timeit(lambda: search(current_list,
middle_element), number=10)
            file.write(str(time_search) + '\n')
        file.close()

        # Получим данные при поиске элемента в списке в худшем случае.
        last_element = -10  # Таких элементов нет в списке
        with open(filename_worst, 'a') as file:
            time_search = timeit.timeit(lambda: search(current_list,
last_element), number=10)
            file.write(str(time_search) + '\n')
        file.close()

        # Увеличим размер данного списка.
        for i in range(1, 1001):
            current_list.append(current_list[-1] + 1)
```

Рисунок 1 – Код программы получения данных о времени поиска.

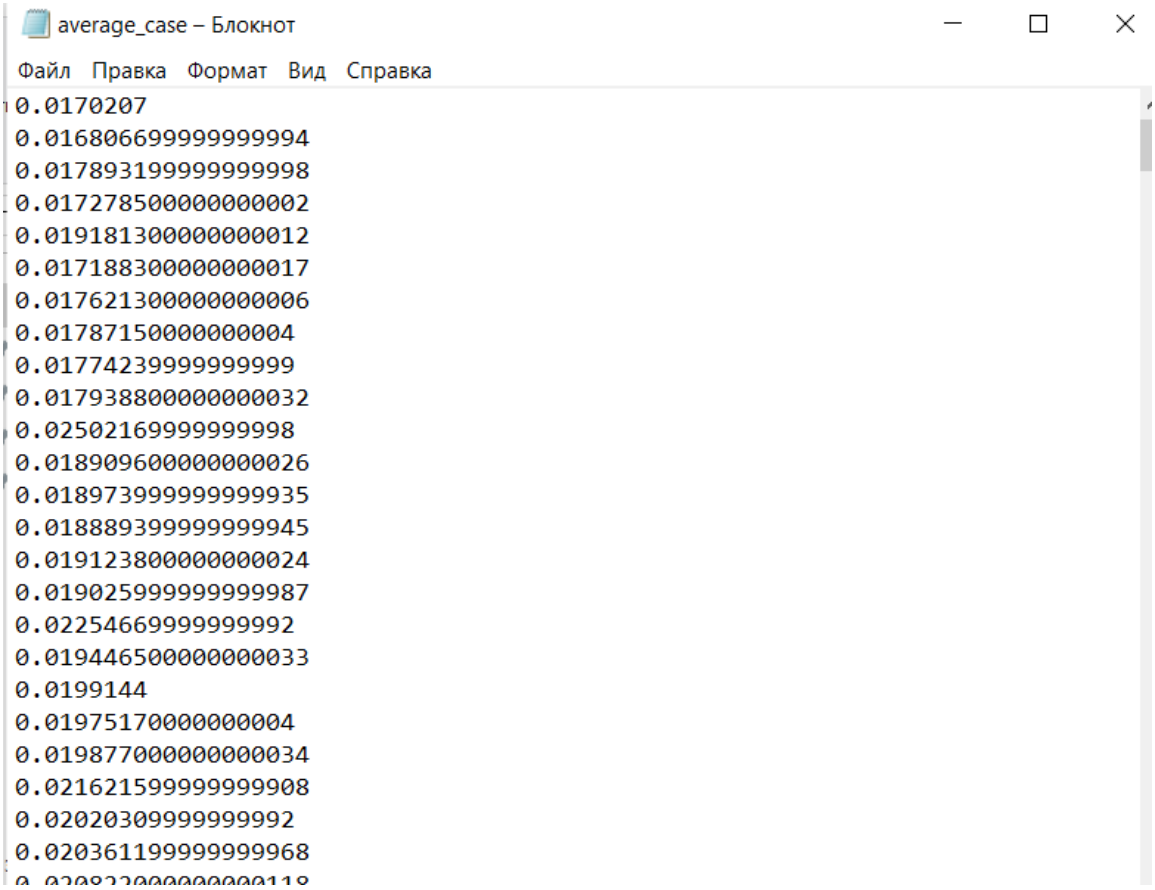

Рисунок 2 – Содержимое файла «average_case.txt».

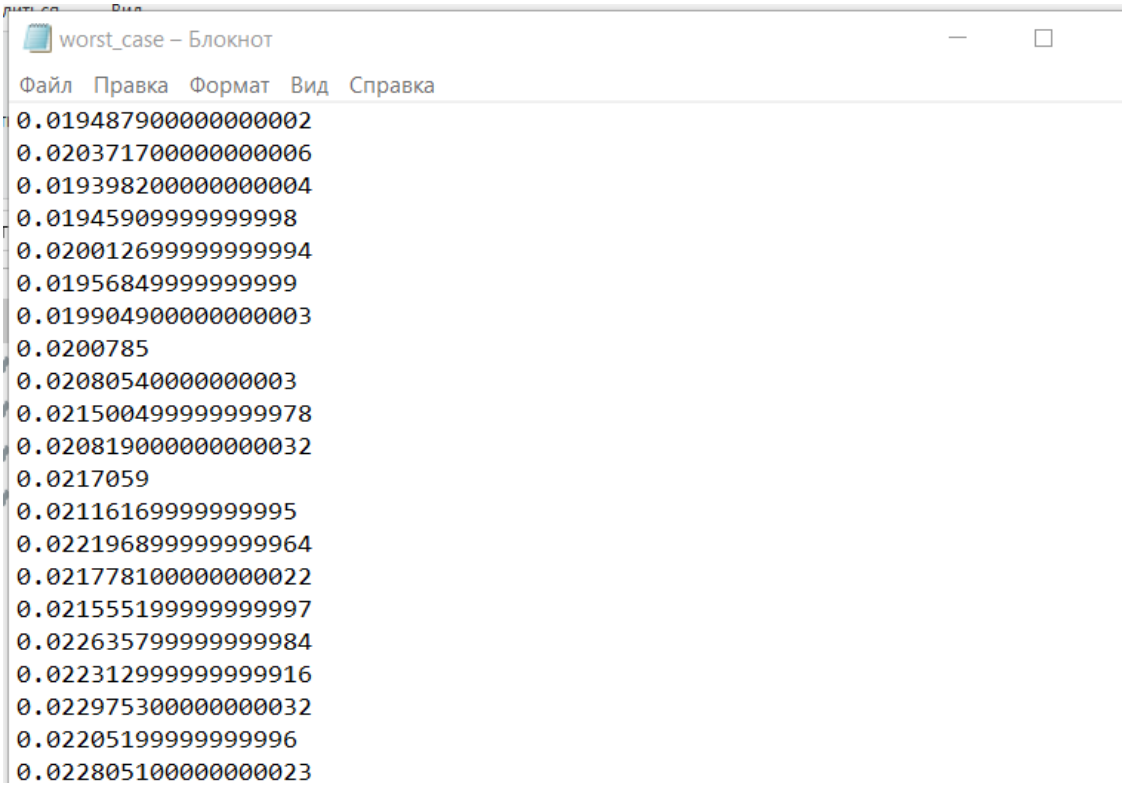

Рисунок 3 – Содержимое файла «worst_case.txt».

График зависимости времени нахождения элемента в списке в среднем случае от размера списка (рис. 4). Коэффициент Пирсона (парной корреляции) в данном случае = 0,991819629 (близок к единице).

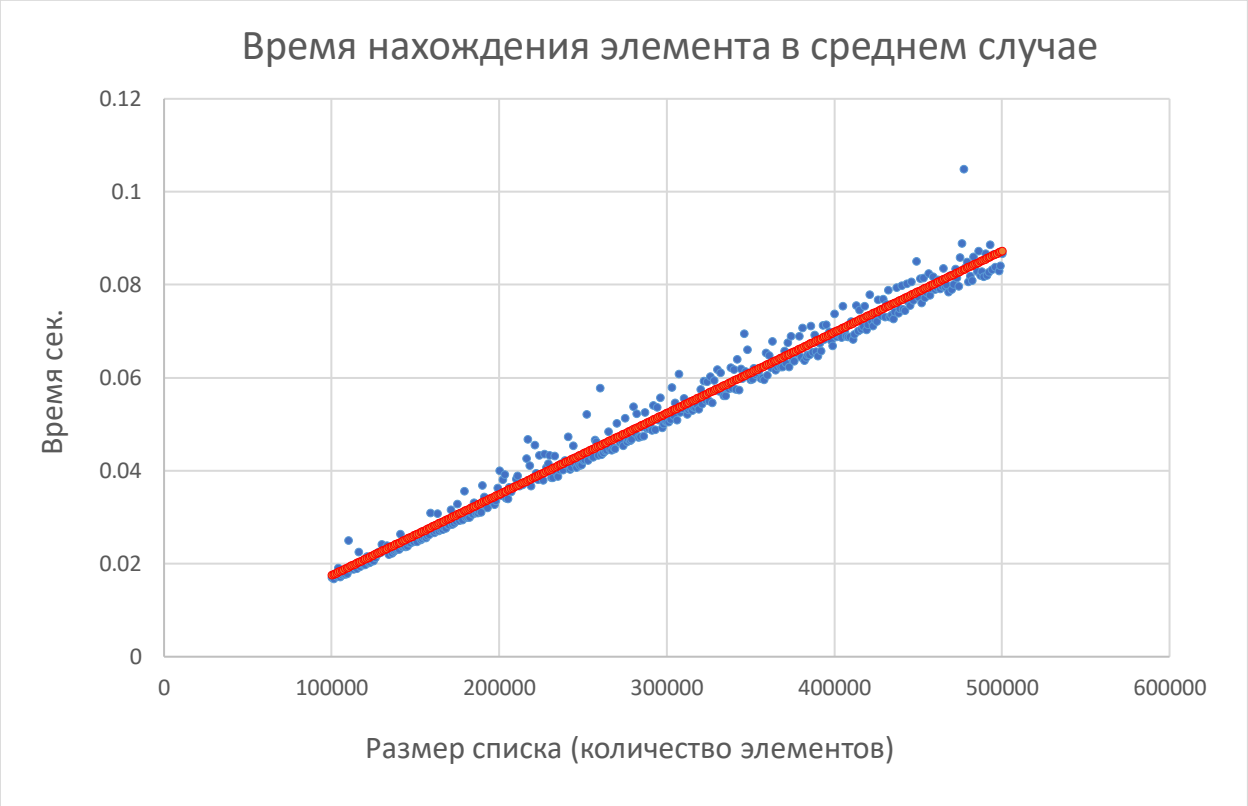

Рисунок 4 – Точечный график зависимости времени поиска в среднем случае от размера списка.

График зависимости времени нахождения элемента в списке в худшем случае от размера списка (рис. 5). Коэффициент Пирсона (парной корреляции) в данном случае = 0,992797253 (также близок к единице).

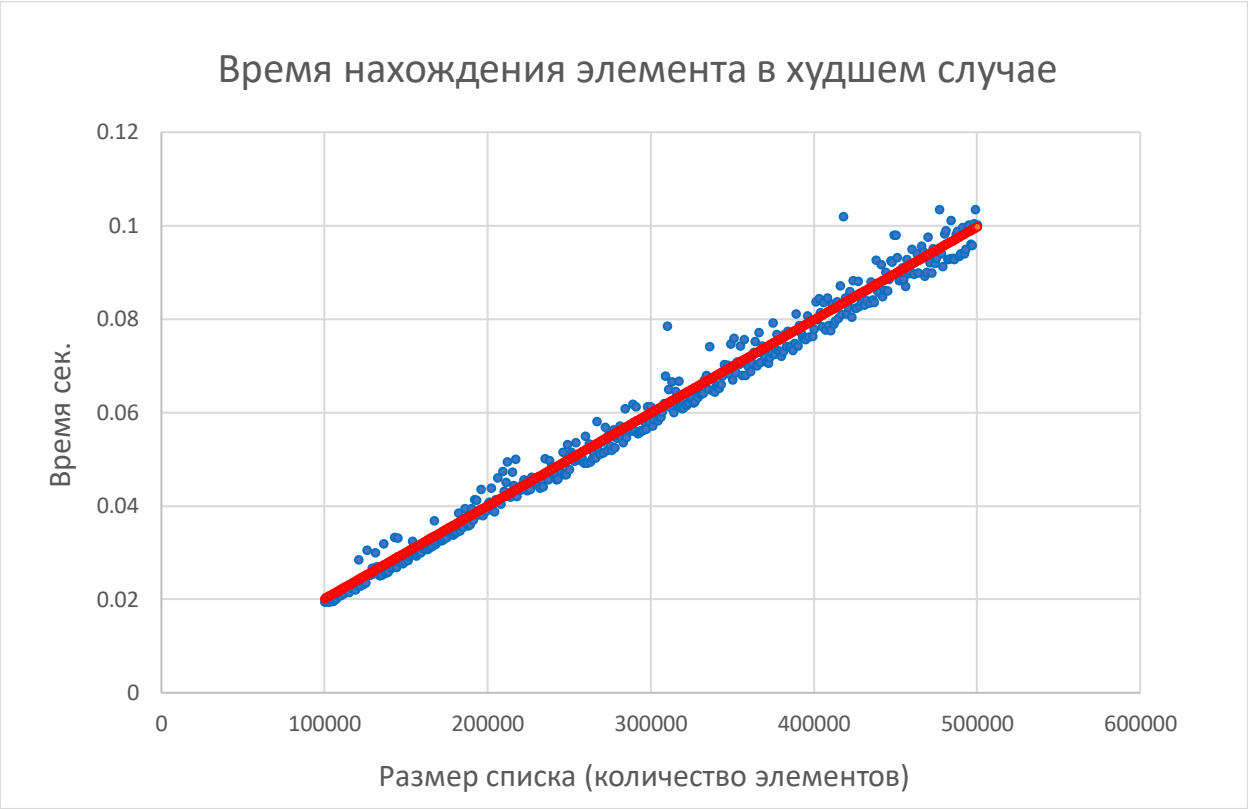

Рисунок 5 – Точечный график зависимости времени поиска в среднем случае от размера списка.

Вывод: в ходе выполнения лабораторной работы был изучен алгоритм линейного поиска, были проведены исследования зависимости времени работы алгоритма от размера списка в среднем и худшем случае. Время линейно зависит от размера списка. Также был более подробно изучен метод наименьших квадратов для нахождения коэффициентов линейного уравнения и коэффициент парной корреляции (коэффициент Пирсона). Были получены коэффициенты парной корреляции для обоих случаев нахождения элемента в списке, в обоих случаях коэффициент был близок к единице, это говорит о тесной взаимосвязи одной переменной от другой.