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 ЛАБОРАТОРНОЙ РАБОТЕ №3
дисциплины «Алгоритмизация»

Выполнил:
Епифанов Алексей Александрович
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной
техники и автоматизированных систем
», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Роман Александрович
_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Порядок выполнения работы:

1. Написал программу, которая выводит в виде графика зависимость времени выполнения функции линейного поиска от размера массива, я рассмотрел 2 случая: средний и худший, и соответственно моя программа вывела 2 графика. Также на графике есть прямая, построенная по методу меньших квадратов, а в консоль выводится коэффициент парной корреляции.

main.py × Параметры

program > main.py > ...

```python
import random as rnd
import matplotlib.pyplot as plt
import numpy as np
import timeit


def find(a, b, len):
    for i in range(len):
        if b == a[i]:
            return i
    return -1


def create_graph(b, c, aur, bur, namegraph):
    plt.scatter(b, c, s=5)
    y_line = aur * np.array(b) + bur
    plt.plot(b, y_line, color='red')
    plt.title(namegraph + " случай")
    plt.xlabel("Размер массива")
    plt.ylabel("Время работы функции")
    correlation_coefficient = np.corrcoef(c, b)[0, 1]
    return correlation_coefficient


correlation_v = []
# Цикл нужен для создания двух графиков, один при среднем случае, второй при худшем
for namegraph in ["Средний", "Худший"]:
    x = [i for i in range(10, 10001, 10)]
    time = []
    x2 = []
    xtime = []
    randmax = 1000000
    for i in x:
        a = [rnd.randint(1, randmax) for j in range(i)]
        if namegraph == "Средний":
            b = a[rnd.randint(1, len(a)-1)]
        else:
            b = randmax+1
        timer = (timeit.timeit(lambda: find(a, b, i), number=50))/50
        time.append(timer)

    # Вычисление коэффицентов в системе уравнений метода наименьших квадратов
    sx = sum(x)
    stime = sum(time)
    sx2 = sum(i**2 for i in x)
    sxtime = sum(i*j for i, j in zip(x, time))
    n = len(x)
    # k - это коэффициент, при котором вычитание
    # из первого уравнения второго,
    # умноженного на него, приводит к нулю в коэффициенте при x.
    # Таким образом, мы сможем вычислить свободный коэффициент.
    k = sx2/sx
    # bur - это свободный коэффицент
    bur = (sxtime - k*stime)/(sx-k*n)
    # aur - это коэффицент при x
    aur = (stime - bur*n)/sx
    # Создание графических окон
    plt.figure(namegraph)
    plt.subplots_adjust(left=0.2)

    # Создание графиков
    correlation_v.append(create_graph(x, time, aur, bur, namegraph))

print("Коэффициент корреляции в первом случае =",
      correlation_v[0], "\na во втором случае =", correlation_v[1])

# Показ графиков
plt.show()
```

Рисунок 1. Код программы

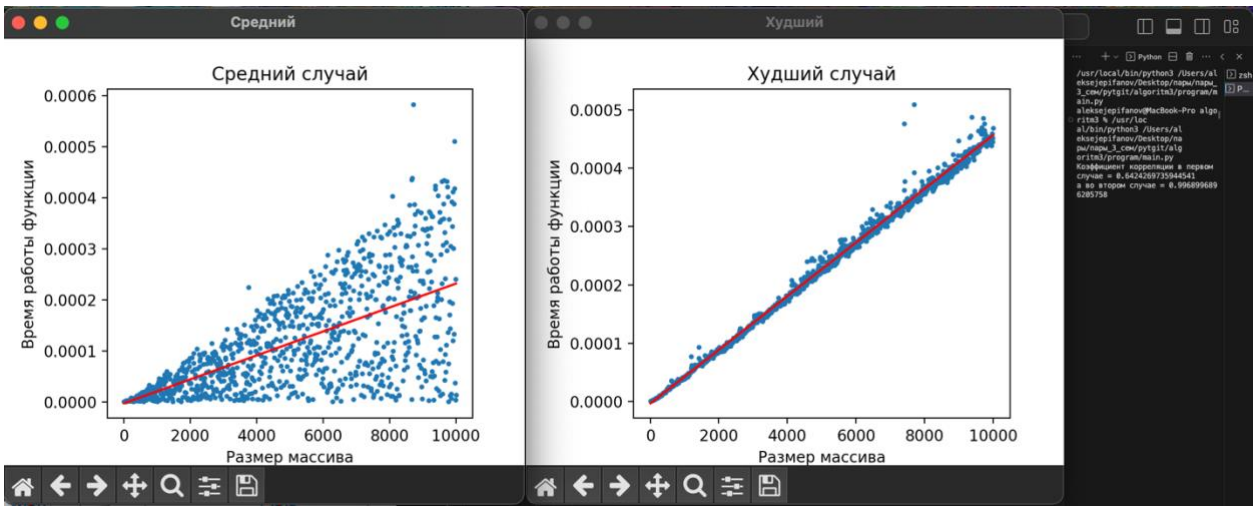

Рисунок 2. Вывод графиков и вывод в консоли

В ходе выполнения лабораторной работы был проведен анализ зависимости времени выполнения функции линейного поиска от размера массива в двух случаях: среднем и худшем. Из полученных результатов можно сделать следующий вывод: время работы функции в худшем случае линейно зависимо от размера массива, тогда как время работы функции в среднем случае случайно, но, практически всегда, не превосходит времени, затраченного на выполнение функции в худшем случае.