

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 X Параметры
program > main.py > ...
1 import random as rnd
2 import matplotlib.pyplot as plt
3 import numpy as np
4 import timeit
5
6
7 def find(a, b, len):
8     for i in range(len):
9         if b == a[i]:
10             return i
11     return -1
12
13
14 def create_graph(b, c, aur, bur, namegraph):
15     plt.scatter(b, c, s=5)
16     y_line = aur * np.array(b) + bur
17     plt.plot(b, y_line, color='red')
18     plt.title(namegraph + " случай")
19     plt.xlabel("Размер массива")
20     plt.ylabel("Время работы функции")
21     correlation_coefficient = np.corrcoef(c, b)[0, 1]
22     return correlation_coefficient
23
24
25 correlation_v = []
26 # Цикл нужен для создания двух графиков, один при среднем случае, второй при худшем
27 for namegraph in ["Средний", "Худший"]:
28     x = [i for i in range(10, 10001, 10)]
29     time = []
30     x2 = []
31     xtime = []
32     randmax = 1000000
33     for i in x:
34         a = [rnd.randint(1, randmax) for j in range(i)]
35         if namegraph == "Средний":
36             b = a[rnd.randint(1, len(a)-1)]
37         else:
38             b = randmax+1
39         timer = (timeit.timeit(lambda: find(a, b, i), number=50))/50
40         time.append(timer)
41
42     # Вычисление коэффициентов в системе уравнений метода наименьших квадратов
43     sx = sum(x)
44     stime = sum(time)
45     sx2 = sum(i**2 for i in x)
46     sxtime = sum(i*j for i, j in zip(x, time))
47     n = len(x)
48     # k - это коэффициент, при котором вычитание
49     # из первого уравнения второго,
50     # умноженного на него, приводит к нулю в коэффициенте при x.
51     # Таким образом, мы сможем вычислить свободный коэффициент.
52     k = sx2/sx
53     # bur - это свободный коэффициент
54     bur = (sxtime - k*stime)/(sx-k*n)
55     # aur - это коэффициент при x
56     aur = (stime - bur*n)/sx
57     # Создание графических окон
58     plt.figure(namegraph)
59     plt.subplots_adjust(left=0.2)
60
61     # Создание графиков
62     correlation_v.append(create_graph(x, time, aur, bur, namegraph))
63
64 print("Коэффициент корреляции в первом случае =",
65       correlation_v[0], "\nа во втором случае =", correlation_v[1])
66
67 # Показ графиков
68 plt.show()
69

```

Рисунок 1. Код программы

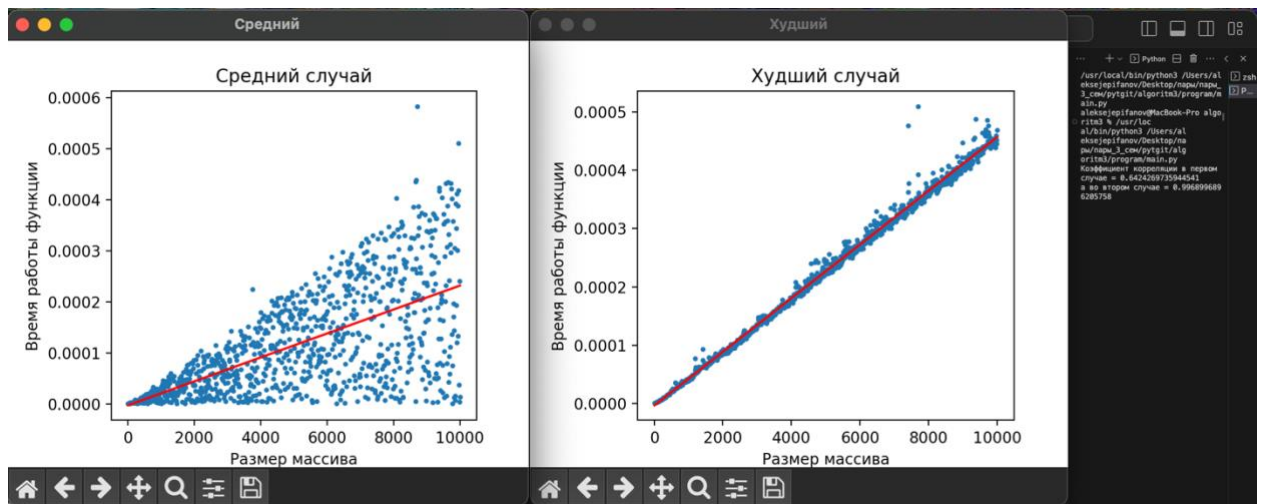


Рисунок 2. Вывод графиков и вывод в консоли

В результате выполнения работы можно сделать вывод: время работы среднего случая более случайно, чем время работы худшего случая, но в общем в обоих случаях время растет с ростом размера массива.