

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 №4
дисциплины
«Алгоритмизация»

Выполнил:
Ибрагимов Муса Айнудинович
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:
Воронкин Роман Александрович

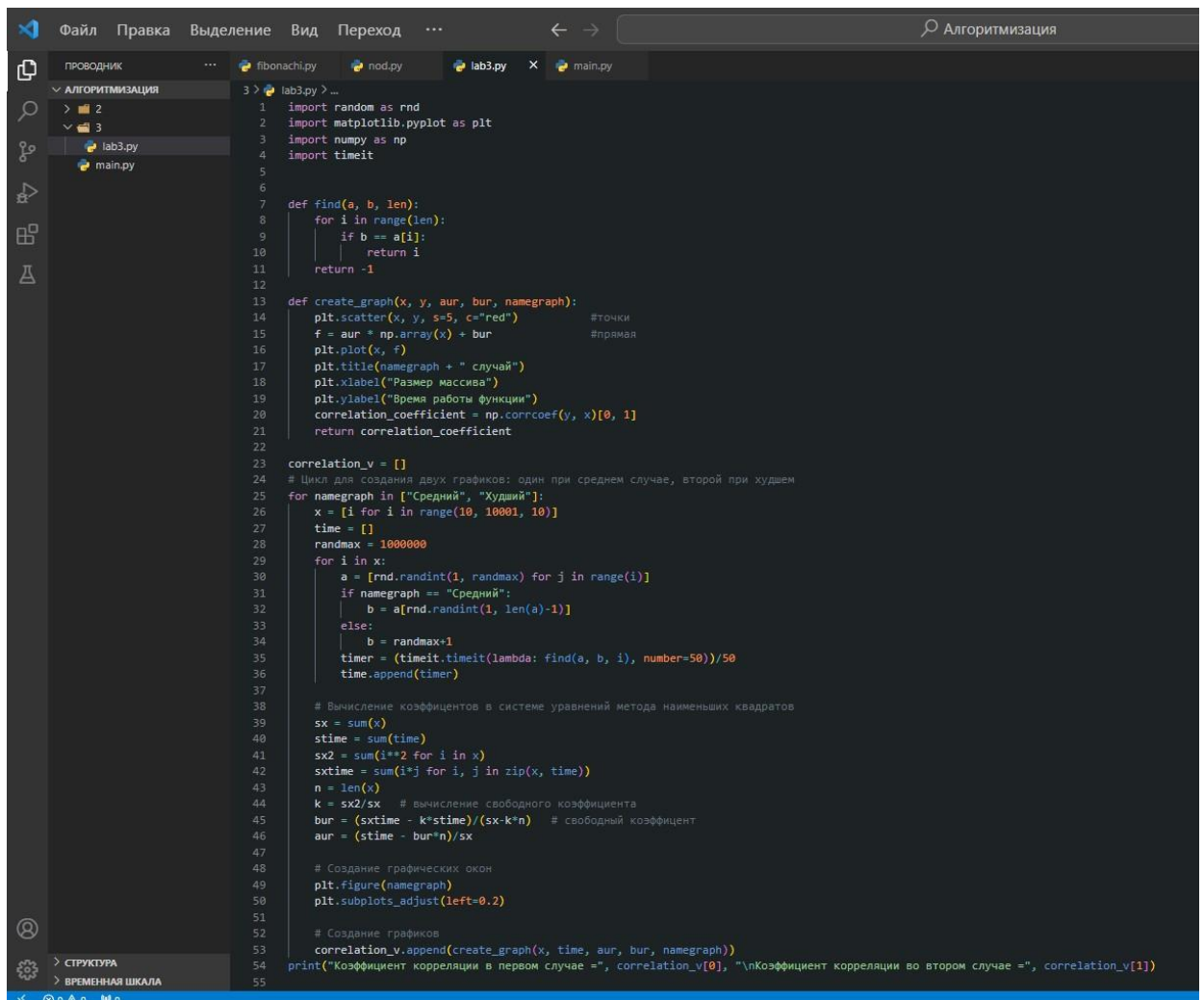
(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2024 г.

Порядок выполнения работы:

1. Написал программу, которая строит график зависимости времени выполнения функции линейного поиска от размера массива, я рассмотрел 2 случая: средний и худший, и соответственно программа вывела 2 графика. Кроме того, на графике присутствует прямая, построенная методом наименьших квадратов, и в выводе консоли присутствует коэффициент парной корреляции:



```
Файл  Правка  Выделение  Вид  Переход  ...  Алгоритмизация
проектор
ALГОРИТМИЗАЦИЯ
2
3
lab3.py
main.py
3 > lab3.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 def create_graph(x, y, aur, bur, namegraph):
14     plt.scatter(x, y, s=5, c='red') # точки
15     f = aur * np.array(x) + bur # прямая
16     plt.plot(x, f)
17     plt.title(namegraph + " случай")
18     plt.xlabel("Размер массива")
19     plt.ylabel("Время работы функции")
20     correlation_coefficient = np.corrcoef(y, x)[0, 1]
21     return correlation_coefficient
22
23 correlation_v = []
24 # Цикл для создания двух графиков: один при среднем случае, второй при худшем
25 for namegraph in ["Средний", "Худший"]:
26     x = [i for i in range(10, 10001, 10)]
27     time = []
28     randmax = 1000000
29     for i in x:
30         a = [rnd.randint(1, randmax) for j in range(i)]
31         if namegraph == "Средний":
32             b = a[rnd.randint(1, len(a)-1)]
33         else:
34             b = randmax+1
35         timer = (timeit.timeit(lambda: find(a, b, i), number=50))/50
36         time.append(timer)
37
38 # Вычисление коэффициентов в системе уравнений метода наименьших квадратов
39 sx = sum(x)
40 stime = sum(time)
41 sx2 = sum(i**2 for i in x)
42 sxtime = sum(i*j for i, j in zip(x, time))
43 n = len(x)
44 k = sx2/sx # вычисление свободного коэффициента
45 bur = (sxtime - k*stime)/(sx-k*n) # свободный коэффициент
46 aur = (stime - bur*n)/sx
47
48 # Создание графических окон
49 plt.figure(namegraph)
50 plt.subplots_adjust(left=0.2)
51
52 # Создание графиков
53 correlation_v.append(create_graph(x, time, aur, bur, namegraph))
54 print("Коэффициент корреляции в первом случае =", correlation_v[0], "\nКоэффициент корреляции во втором случае =", correlation_v[1])
55
```

Рисунок 1. Код программы

2. Результат выполнения программы:

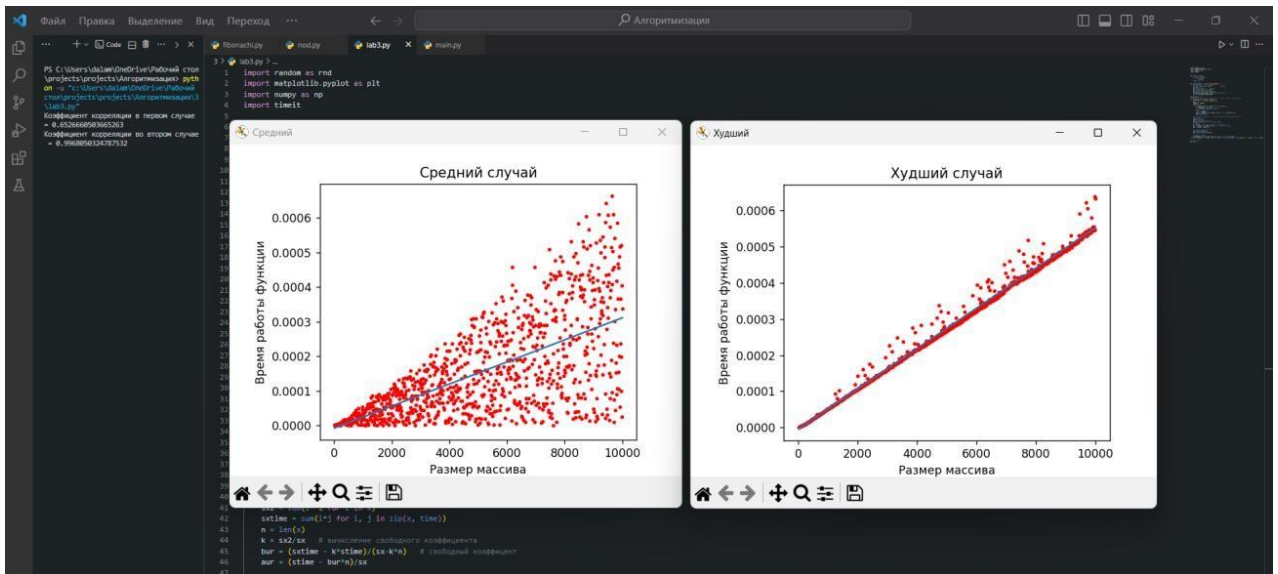


Рисунок 2. Вывод графиков.

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 зависит от размера массива, тогда как время работы функции в среднем случае, практически всегда, не превосходит времени, затраченного на выполнение функции в худшем случае.

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 зависит от размера массива, тогда как время работы функции в среднем случае, практически всегда, не превосходит времени, затраченного на выполнение функции в худшем случае.