

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 №9
дисциплины «Алгоритмизация»

Выполнил:
Ибрагимов Муса Айнуудинович
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 Р.А., доцент кафедры
инфокоммуникаций

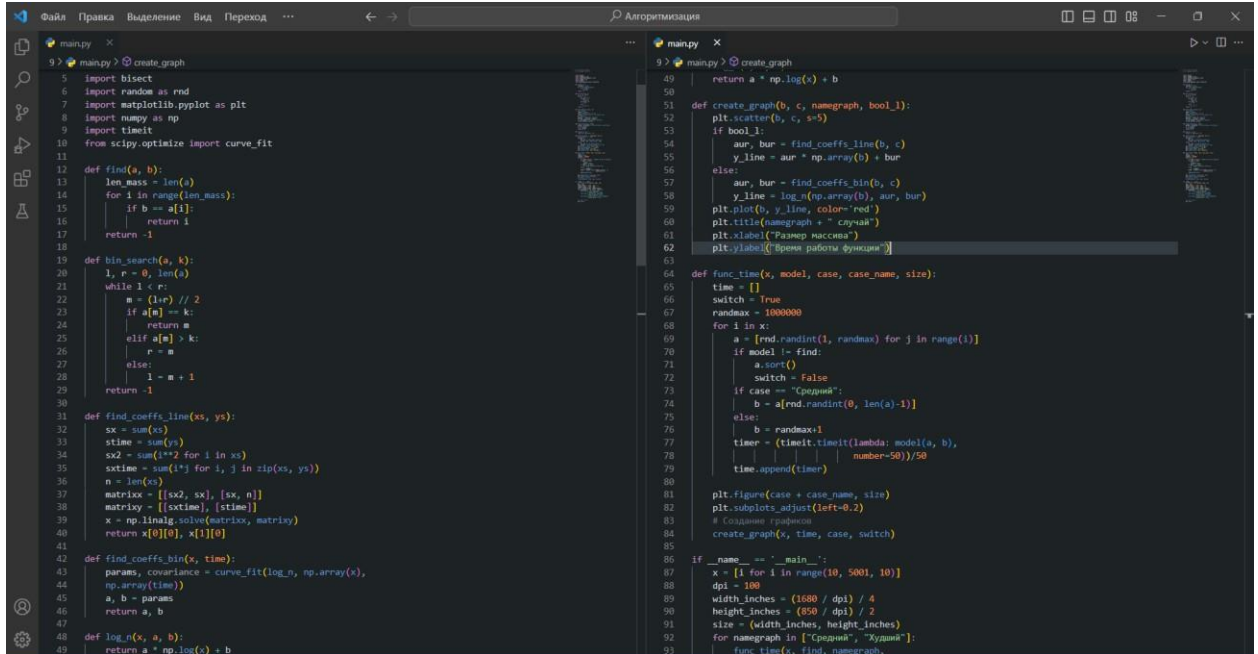
(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2024 г.

Порядок выполнения работы:

1. Написал программу сравнения функций поиска, написанных мной: линейный поиск и бинарный поиск, а так же эти функции сравнил со встроенной в Python функцией бинарного поиска bisect:



```
9 > main.py > create_graph
5 import bisect
6 import random as rnd
7 import matplotlib.pyplot as plt
8 import numpy as np
9 import timeit
10 from scipy.optimize import curve_fit
11
12 def find(a, b):
13     len_mass = len(a)
14     for i in range(len_mass):
15         if b == a[i]:
16             return i
17     return -1
18
19 def bin_search(a, k):
20     l, r = 0, len(a)
21     while l < r:
22         m = (l+r) // 2
23         if a[m] == k:
24             return m
25         elif a[m] > k:
26             r = m
27         else:
28             l = m + 1
29     return -1
30
31 def find_coeffs_line(xs, ys):
32     sx = sum(xs)
33     stime = sum(ys)
34     sx2 = sum(x**2 for i, x in xs)
35     sxtime = sum(x*y for i, j in zip(xs, ys))
36     n = len(xs)
37     matrix = [[sx2, sx], [sx, n]]
38     matrixy = [sxtime, stime]
39     x = np.linalg.solve(matrix, matrixy)
40     return x[0][0], x[1][0]
41
42 def find_coeffs_bin(x, time):
43     params, covariance = curve_fit(log_n, np.array(x),
44     np.array(time))
45     a, b = params
46     return a, b
47
48 def log_n(x, a, b):
49     return a * np.log(x) + b
50
51 def create_graph(b, c, namegraph, bool_1):
52     plt.scatter(b, c, s=5)
53     if bool_1:
54         aur, bur = find_coeffs_line(b, c)
55         y_line = aur * np.array(b) + bur
56     else:
57         aur, bur = find_coeffs_bin(b, c)
58         y_line = log_n(np.array(b), aur, bur)
59     plt.plot(b, y_line, color='red')
60     plt.title(namegraph + " - лучший")
61     plt.xlabel("Размер массива")
62     plt.ylabel("Время работы функции")
63
64 def func_time(x, model, case, case_name, size):
65     time = []
66     switch = True
67     randmax = 1000000
68     for i in x:
69         a = [rnd.randint(1, randmax) for j in range(i)]
70         if model != find:
71             a.sort()
72             switch = False
73         if case == "Средний":
74             b = a[rnd.randint(0, len(a)-1)]
75         else:
76             b = randmax-1
77         timer = (timeit.timeit(lambda: model(a, b),
78             number=50))/50
79         time.append(timer)
80
81 plt.figure(case + case_name, size)
82 plt.subplots_adjust(left=0.2)
83 # Создание графиков
84 create_graph(x, time, case, switch)
85
86 if __name__ == '__main__':
87     x = [i for i in range(10, 5001, 10)]
88     dpi = 100
89     width_inches = (1680 / dpi) / 4
90     height_inches = (1050 / dpi) / 2
91     size = (width_inches, height_inches)
92     for namegraph in ["Худший", "Худшая"]:
93         func_time(x, find, namegraph,
```

Рисунок 1. Код программы

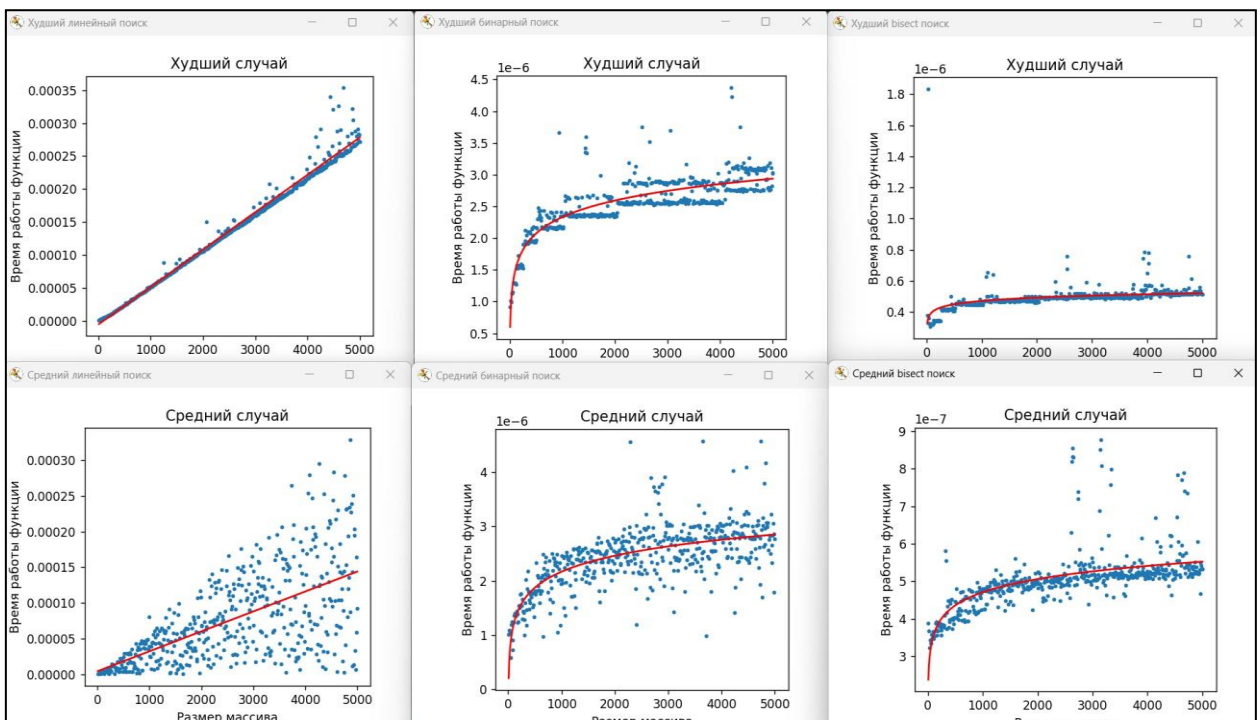


Рисунок 2. Результат работы программы

Вывод: в ходе выполнения лабораторной работы были исследованы функции поиска в двух случаях: среднем и худшем. Лучшей оказалась функция, встроенная в Python: `bisect.bisect_left()`, а худшей – линейный поиск.