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 №5
дисциплины «Алгоритмизация»

Выполнил:
Середа Кирилл Витальевич
1 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника», очная
форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Роман Александрович

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Тема: Аналитика Пузырьковой сортировки

Ход выполнения заданий

1) Написал программу для аналитики пузырьковой сортировки и построил графики

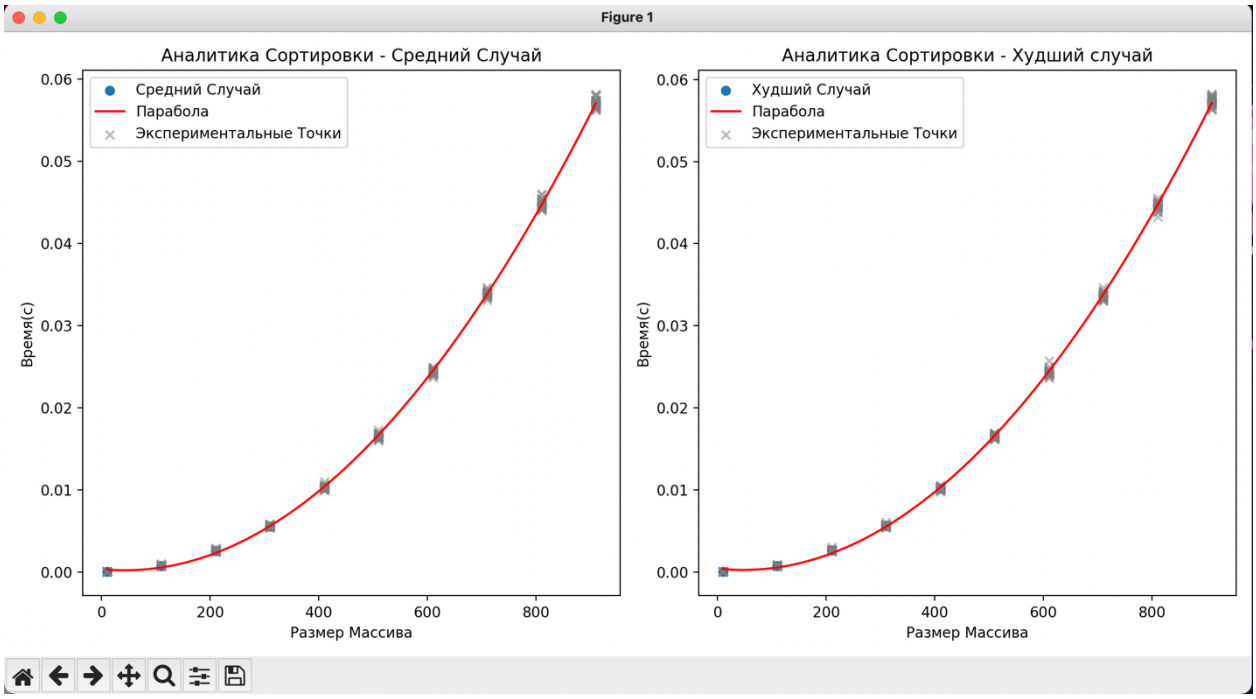

Рисунок 1 – графики аналитики

```
Линейная зависимость для худшего случая: 7.884220112373775e-08a + -7.809560708586186e-06b + 0.00044063824545840667c
Линейная зависимость для среднего случая: 7.832629843434357e-08a + -7.491411709090998e-06b + 0.0004153284666414396c
Средний Коэффициент Корреляции Для Обоих Случаев: 0.96
```

Рисунок 2 – Коэффициент корреляции и линейная зависимость

Код программы:

```
import random
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import pearsonr
import timeit
from tqdm import tqdm

def bubble_sort(arr):
    n = len(arr)
    for i in range(n):
        for j in range(0, n-i-1):
            if arr[j] > arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr[j]

def rndm_arr(size):
    return [random.randint(1, 1000) for _ in range(size)]

def anlzBubble(num_experiments, max_array_size):
    sizes = list(range(10, max_array_size + 1, 100))
    avg_results = []
    worst_results = []
    all_results = []
```

```
    for size in sizes:
        random_arrays = [rndm_arr(size) for _ in range(num_experiments)]
        avg_times = []
        worst_times = []
        for i, arr in enumerate(tqdm(random_arrays, desc=f"Размер массива:
{size}")):
            avg_arr = arr.copy()
            worst_arr = arr.copy()[::-1]

            avg_time = timeit.timeit(lambda: bubble_sort(avg_arr), number=1)
            avg_times.append(avg_time)

            worst_time = timeit.timeit(lambda: bubble_sort(worst_arr),
number=1)
            worst_times.append(worst_time)

            all_results.append((size, avg_time, worst_time))

        avg_results.append(np.mean(avg_times))
        worst_results.append(np.mean(worst_times))

    return sizes, avg_results, worst_results, all_results

def plot_results(sizes, avg_case_results, worst_results, all_results):
    plt.figure(figsize=(12, 6))

    # Средний случай
    plt.subplot(1, 2, 1)
    plt.scatter(sizes, avg_case_results, label="Средний Случай", marker='o')
    plt.xlabel("Размер Массива")
    plt.ylabel("Время(с)")
    plt.title("Аналитика Сортировки - Средний Случай")

    # Рассчеты и построение графика
    avg_fit = np.polyfit(sizes, avg_case_results, 2)
    avg_curve = np.poly1d(avg_fit)
    x_curve = np.linspace(min(sizes), max(sizes), 100)
    plt.plot(x_curve, avg_curve(x_curve), label="Парабола", color='r')

    # Вывод мн-ва точек экспериментов
    sizes_points, avg_points, worst_points = zip(*all_results)
    plt.scatter(sizes_points, avg_points, color='gray', alpha=0.5,
marker='x', label="Экспериментальные Точки")

    plt.legend()

    # Худший случай
    plt.subplot(1, 2, 2)
    plt.scatter(sizes, worst_results, label="Худший Случай", marker='o')
    plt.xlabel("Размер Массива")
    plt.ylabel("Время(с)")
    plt.title("Аналитика Сортировки - Худший случай")

    # Рассчеты и построение графика
    worst_fit = np.polyfit(sizes, worst_results, 2)
    worst_curve = np.poly1d(worst_fit)
    x_curve = np.linspace(min(sizes), max(sizes), 100)
    plt.plot(x_curve, worst_curve(x_curve), label="Парабола", color='r')

    # Вывод мн-ва точек экспериментов
    plt.scatter(sizes_points, worst_points, color='gray', alpha=0.5,
marker='x', label="Экспериментальные Точки")

    plt.legend()
```

```
    plt.tight_layout()
    plt.show()
    print(f"Линейная зависимость для худшего случая: {worst_fit[0]}a +
{worst_fit[1]}b + {worst_fit[2]}c")
    print(f"Линейная зависимость для среднего случая: {avg_fit[0]}a +
{avg_fit[1]}b + {avg_fit[2]}c")
if __name__ == "__main__":
    num_experiments = 30  # Количество экспериментов
    max_array_size = 1000  # Предельный размер массивов
    sizes, avg_results, worst_results, all_results =
anlzBubble(num_experiments, max_array_size)
    plot_results(sizes, avg_results, worst_results, all_results)

    # Рассчет коэффициента
    cor_coef = pearsonr(sizes, avg_results)
    print(f"Средний Коэффициент Корреляции Для Обоих Случаев:
{cor_coef[0]:.2f}")
```

Вывод: в ходе выполнения работы было выявлено, что алгоритм пузырьковой сортировки достаточно быстро замедляется и становится не эффективным