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 ПРАКТИЧЕСКОЙ РАБОТЕ №3
дисциплины «Алгоритмизация»
Вариант___

Выполнил:
Репкин Александр Павлович
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., канд. техн. наук,
доцент кафедры инфокоммуникаций

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

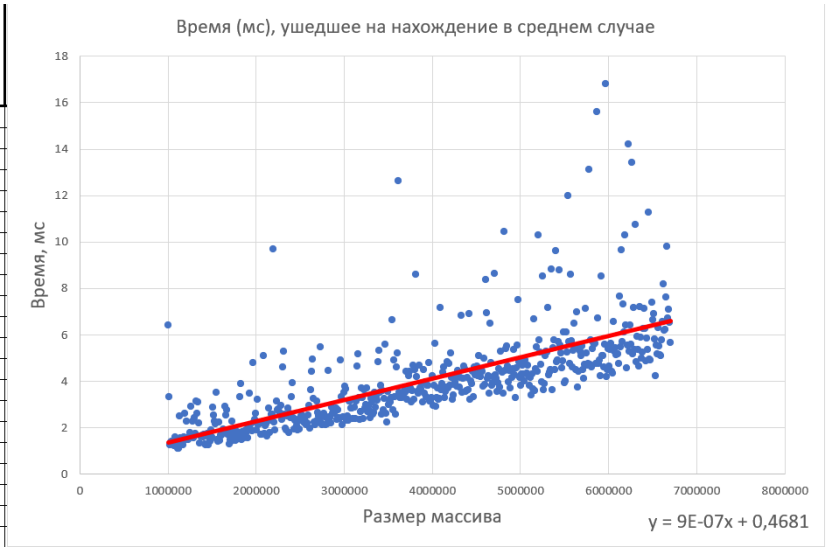
Порядок выполнения работы:

1. Создана программа на основе поставленного задания – необходимо проанализировать скорость нахождения элемента в списке в зависимости от его положения в нём. Требовалось рассмотреть два варианта – худший (Элемента нет в списке), средний (Элемент находится недалеко от середины списка). На основе полученных данных о затраченном времени необходимо построить график, используя метод наименьших квадратов.

```
public static void main(String[] args) {
    Random rand = new Random();
    while (needed <= 6700000) {
        List<Integer> numbers = new ArrayList<>();
        results.append(needed).append(";");
        for(int i = 0; i < needed; i++) numbers.add(i);
        int max = (int) (needed * 0.6);
        int min = (int) (needed * 0.4);
        // Middle
        int randomNumber = rand.nextInt( bound: (max - min) + 1) + min;
        long startTime = System.nanoTime();
        for (Integer integer : numbers) {
            if (integer == randomNumber) {
                //System.out.println(j);
                break;
            }
        }
        long endTime = System.nanoTime();
        long spent_time = (endTime - startTime);  //divide by 1000000 to get milliseconds.
        results.append(spent_time/ 1000000).append(",").append((spent_time/ 10000)%100).append(";");
        //System.out.println("Spent time is " + spent_time / 10000);
        //System.out.println("\n-1\n");
        // Doesn't exist, -1
        startTime = System.nanoTime();
        for (Integer number : numbers) {
            if (number == needed) {
                System.out.println("Oh no! Why is it here?!");
            }
        }
        endTime = System.nanoTime();
        spent_time = (endTime - startTime);  //divide by 1000000 to get milliseconds.
        results.append(spent_time/ 1000000).append(" ").append((spent_time/ 10000)%100).append("\n");
```

Рисунок 1. Полученный код.

Длина массива	Время (мс), ушедшее на нахождение в среднем случае
1000000	6,41
1010000	3,34
1020000	1,25
1030000	1,38
1040000	1,41
1050000	1,29
1060000	1,28
1070000	1,22
1080000	1,6
1090000	1,2
1100000	1,42
1110000	1,1
1120000	1,11
1130000	2,51
1140000	1,57
1150000	1,51
1160000	1,34
1170000	1,26
1180000	1,58
1190000	2,62
1200000	1,47

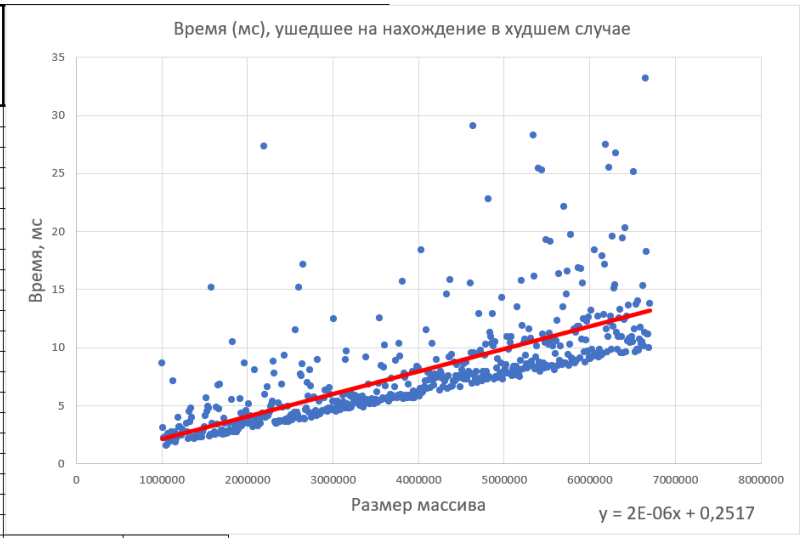


Коэффициент парной корреляции

0,71675155

Рисунок 2. Полученный график для среднего случая.

Длина массива	Время (мс), ушедшее на нахождение в худшем случае
1000000	8,68
1010000	3,13
1020000	2,2
1030000	2,19
1040000	2,33
1050000	1,58
1060000	1,6
1070000	2,6
1080000	1,98
1090000	2,5
1100000	1,94
1110000	2,71
1120000	2,23
1130000	7,14
1140000	2,49
1150000	2,75
1160000	1,87
1170000	2,13
1180000	2,47
1190000	3,99
1200000	3,16



Коэффициент парной корреляции

0,69787993

Рисунок 3. Полученный график для худшего случая.

Вывод: в ходе выполнения практической работы, была рассмотрена зависимость затраты времени на поиск элемента в списке в зависимости от размера списка и положения элемента в нём. Согласно полученным данным, зависимость поиска элемента в массиве – линейная.