Лабораторная работа 1. Поиск.

Постановка задачи

Перед выполнением лабораторных работ прочитать файл *README.pdf*, содержащий общий исходный код, скрипты и рекомендации для выполнения лабораторных работ, а также дополнительную информацию.

Реализовать алгоритмы линейного поиска, бинарного поиска и экспоненциального поиска (*galloping search*, *exponential search*, *«поиск от края»*). В качестве ключа использовать целое число (*int*).

Проверка требуемого количества оперативной памяти:

```
<количество данных> * <тип данных>
Массив:
    максимальное количество элементов 5000000
    тип данных int
Сортировка слиянием:
    требует по памяти n
Максимальное количество элементов для поиска:
    1000
    тип данных int
Суммарно:
    5000000 * 4 * 2 + 1000 * 4 =
    40004000 байт = 39 МБ
```

Экспериментальное исследование

- Память выделяется функцией *malloc*:
  `int* arr = (int*)malloc(sizeof(int) * n);`
- Массивы заполнить псевдослучайными числами с равномерным распределением из интервала [0, 10 000 000].
- Необходимо измерить время работы каждого алгоритма при различном количестве элементов в массиве — заполните таблицу 1 для каждого алгоритма, построить графики по полученной таблице (пример на рисунках 1, 2)
- Перед выполнением бинарного и экспоненциального поиска отсортировать массив сортировкой слиянием. Для заполнения таблицы 1 не учитывать время работы сортировки.
- По результатам экспериментов определите, какой алгоритм работает быстрее и почему (таблица 1).

- Для заполнения таблицы 2 измерить суммарное время выполнения поиска указанного количества случайных элементов для различного количества элементов в массиве. Сортировка выполняется 1 раз.
- Определить приблизительный порог «окупаемости» накладных расходов на сортировку при выполнении бинарного поиска.

Таблица 1. Время выполнения поиска элемента в массиве

#	Количество элементов в массиве	Время выполнения линейного поиска, мкс	Время выполнения бинарного поиска, мкс	Время выполнения экспоненциального поиска, мкс
1	250 000			
2	500 000			
3	750 000			
...	...			
20	5 000 000			

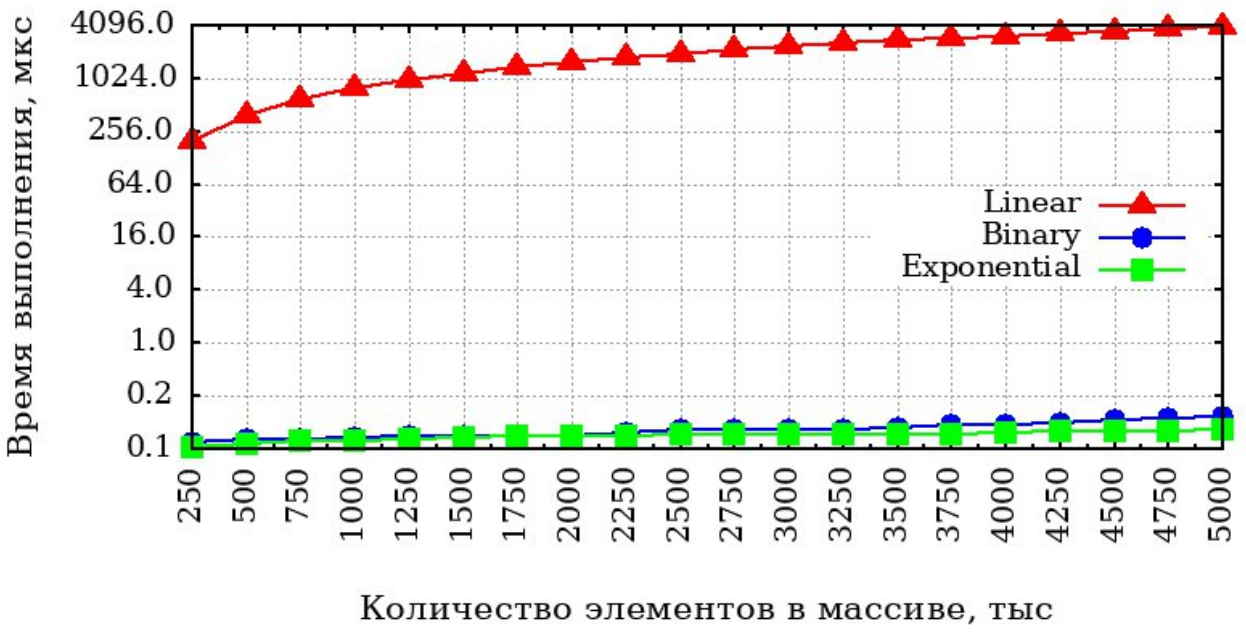

Рисунок 1. Зависимость времени выполнения алгоритмов поиска элемента в массиве от количества элементов в массиве

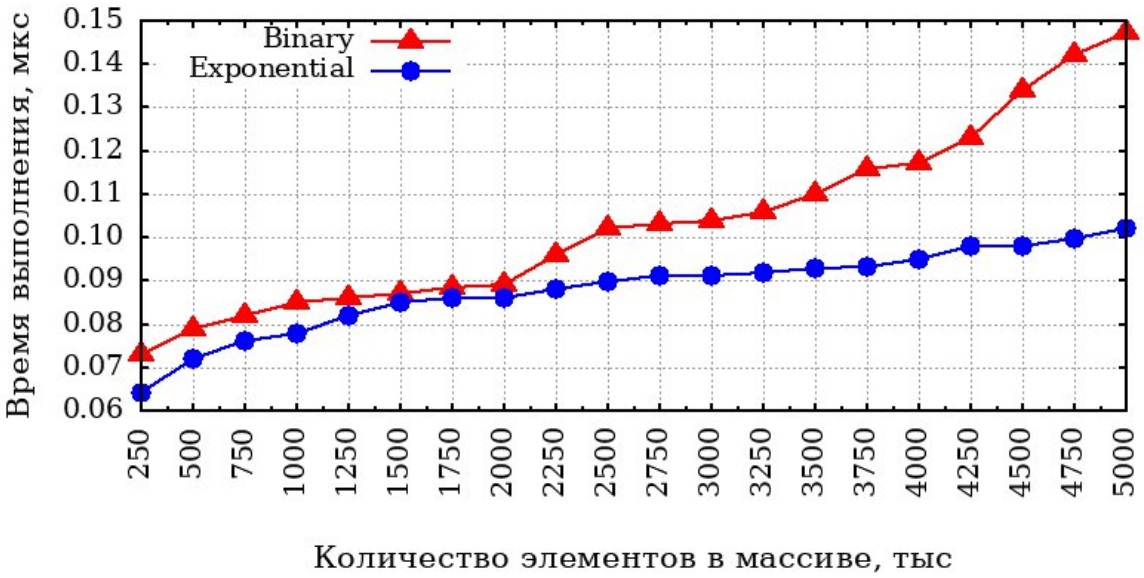

Рисунок 2. Зависимость времени выполнения алгоритмов бинарного и экспоненциального поиска от количество элементов в массиве

Таблица 2. Результаты экспериментов

#	Кол-во элементов в массиве	Кол-во элементов для поиска	Линейный поиск, мкс	Бинарный поиск, мкс	Бинарный поиск с сортировкой, мкс	Сортировка, мкс
1	1 000 000	50				
2	1 000 000	100				
3	1 000 000	150				
4	1 000 000	200				
5	1 000 000	250				
6	1 000 000	300				
7	1 000 000	350				
8	1 000 000	400				
9	1 000 000	450				
10	1 000 000	500				
11	5 000 000	100				
12	5 000 000	200				
...	...	...	...	...	...	
20	5 000 000	1000				

Контрольные вопросы

- Что такое вычислительная сложность?
- Основные классы вычислительной сложности.
- Механизм работы алгоритмов поиска.
- Вычислительная сложность алгоритмов поиска.
- Продемонстрировать работу алгоритмов поиска.
- Определить вычислительную сложность алгоритмов поиска по исходному коду.
- При каком количестве элементов накладные расходы на сортировку массива «окупаются» для алгоритма бинарного поиска?