Contents

[1. Знакомство с алгоритмами 1](#_Toc3816803)

[1.1. Бинарынй поиск 1](#_Toc3816804)

[1.1.1. Логарифм 1](#_Toc3816805)

[1.2. Время выполнения 2](#_Toc3816806)

[1.3. «О-большое» 2](#_Toc3816807)

[1.3.1. Время выполнения алгоритмов растет с разной скоростью 2](#_Toc3816808)

# Знакомство с алгоритмами

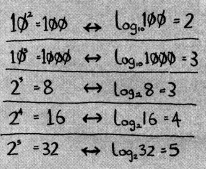
## Бинарынй поиск

**Бинарный поиск** - это алгоритм, на входе он получает отсортированный список элементов. Если элемент, который вы ищете, присутствует в списке, то бинарный поиск возвращает ту позицию, в которой он был найден. В противном случае бинарный поиск возвращает null.

При **бинарном поиске** при каждой итерации исключается половина чисел.

### Логарифм

– сколько раз нужно перемножить n, чтобы получить k.

**Логарифм** – операция, обратная возведению в степерь.

**Примечание**

Бинарный поиск работает только в том случае, если список отсортирован.

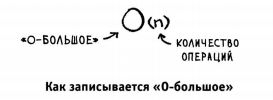
***Например***, имена в телефонной книге хранятся в алфавитном порядке, и вы можете воспользоваться бинарным поиском. А что произойдет, если имена не будут отсортированы?

## Время выполнения

## «О-большое»

**«О-большое»** описывает, насколько быстро работает алгоритм. При этом **«О-большое»** не сообщает скорость в секундах, а позволяет сравнить количество операций. Оно указывает, насколько быстро возрастает время выполнения алгоритма с увеличением размера входных данных.

***Например:***

* имееться список размером **n**. *Простой поиск* должен проверить каждый элемент, поэтому ему придется выполнить **n** операций. Время выполнения **«О-большое»** имеет вид **O(n)**.
* Для проверки списка размером **n** *бинарному поиску* потребуется **log n** итераций. В таком случаее время выполнения **«О-большое»** будет **O(log n).**

### Время выполнения алгоритмов растет с разной скоростью

## «О-большое» определяет время выполнения в худшем случае

При обычном поиске нам известно что О(n). Предположим, нам не пришлось просматривать все записи – мы нашли ее за первую итерацию.

Отработал ли алгоритм за время О(n)? А может, он занял время О(1), потому что результат был получен с первой попытки?

Простой поиск все равно выполняется за время О(n). Просто в данном случае это *лучший* возможный случай. Однако «О-большое» описывает *худший* возможный случай. Фактически мы утверждаем, что в *худшем случае* придется просмотреть каждую запись О(n). И это дает определенные гарантии – вы знаете, что простой поиск никогда не будет работать медленнее O(n).

**Примечание**

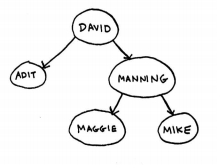
Наряду с *временем худшего случая* также полезно учитывать *среднее время выполнения*.

## Типичные примеры «О-большоего»

## Шпаргалка

* Бинарный поиск работает намного быстрее простого.
* Время выполнения O(log n) быстрее O(n), а с увеличением размера списка, в котором ищется значение, оно становится намного бысрее.
* Скорость алгоритмов не измеряется в секундах.
* Время выполнения алгоритма описывается ростом количества операций.
* Время выполнения алгоритмов выражается как «О-большое».

# Бинарное дерево поиска

**Бинарное дерево** – это дерево в котором для каждого узла все узлы **левог поддерева** содержат *меньшие значения*, а все узлы **правого поддерева** – *большие значения*.

**Идея**: для возможности быстрого поиска елемента, при каждом изменинии содержимого масива, не нужно сортировать его заново.

**Сравнение с бинарным поиском:** в целом процедура поиска *бинарного дерева поиска* напоминает *бинарный поиск*.

Поиск:

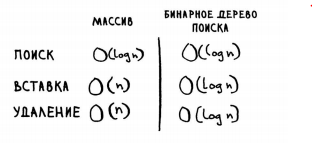
В бинарном дереве поиска: **в среднем** O(log n), **в худшем** O(n).

*бинарным поиском в масиве*: **в худшем** O(log n).

Удаление/Вставка:

В бинарном дереве поиска: **в среднем** O(log n).

Масив: **в среднем** O(n)



# Сортировака выбором