

# 没那么简单的二分法 Binary Search

课程版本 6.1

主讲 令狐冲



扫描二维码关注微信/微博  
获取最新面试题及权威解答

微信: [ninechapter](#)

微博: <http://www.weibo.com/ninechapter>

知乎: <http://zhuankan.zhihu.com/jiuzhang>

官网: <http://www.jiuzhang.com>

# 版权声明

九章的所有课程均受法律保护，不允许录像与传播录像  
一经发现，将被追究法律责任和赔偿经济损失

- 面试中的时间复杂度
- 第一境界：通用模板
  - 递归与非递归的权衡
  - 二分的三大痛点
  - 通用的二分法模板
- 第二境界：叉叉圈圈
  - 找到满足某个条件的第一个位置或者最后一个位置
- 第三境界：保留一半
  - 保留有解的一半，或者去掉无解的一半
- 第四境界：二分答案
  - 二分答案并验证答案偏大还是偏小

<http://www.jiuzhang.com/tutorial/algorithm>

时间复杂度

空间复杂度

$T(N) = T(N/2) + O(1) = O(\log N)$  的推导

$T(N) = T(N/2) + O(N) = O(N)$  的推导

经典二分法原理，改成 First Position / Last Position 之后的代码变化 (video)

什么是递归，用递归如何实现二分法 (video)

内存中栈空间和堆空间的区别

为什么递归可能会造成 Stack Overflow

为什么递归在二分法上，不会造成 Stack Overflow



# 时间复杂度 Time Complexity

$O(N) \Rightarrow$  大概是  $N$  这个数量级的计算

$$O(2N) = O(10N + 1000) = O(1000N) = O(N)$$

$$O(2N^2 + N + 100) = O(N^2)$$

# Time Complexity in Coding Interview

- $O(1)$  极少
- $O(\log n)$  几乎都是二分法
- $O(\sqrt{n})$  几乎是分解质因数
- $O(n)$  高频
- $O(n \log n)$  一般都可能要排序
- $O(n^2)$  数组，枚举，动态规划
- $O(n^3)$  数组，枚举，动态规划
- $O(2^n)$  与组合有关的搜索
- $O(n!)$  与排列有关的搜索

多项式时间复杂度  
**P问题**

非多项式时间复杂度  
**NP问题**

# NP问题的通俗定义

只能用深度优先搜索来解决的问题

# 独孤九剑 —— 破刀式

根据时间复杂度倒推算法是面试常用策略  
如：比 $O(n)$ 更优的时间复杂度只能是 $O(\log n)$ 的二分法



# Recursion or While Loop?

R: Recursion

W: While loop

B: Both work

# Recursion or Non-Recursion

- 面试中是否使用 Recursion 的几个判断条件
  1. 面试官是否要求了不使用 Recursion （如果你不确定，就向面试官询问）
  2. 不用 Recursion 是否会造成实现变得很复杂
  3. Recursion 的深度是否会很深
  4. 题目的考点是 Recursion vs Non-Recursion 还是就是考你是否会 Recursion?
- 记住：不要自己下判断，要跟面试官讨论！
- 面试中通常极少考到 Non-Recursion，考 Recursion 的时候比较多

# 第一境界 二分法模板

<http://www.jiuzhang.com/solutions/binary-search/>

$start + 1 < end$

$start + (end - start) / 2$

$A[mid] ==, <, >$

$A[start] A[end] ? target$

## 二分法常见痛点

- 又死循环了！ what are you 弄撒捏！
- 循环结束条件到底是哪个？
  - $\text{start} \leq \text{end}$
  - $\text{start} < \text{end}$
  - $\text{start} + 1 < \text{end}$
- 指针变化到底是哪个？
  - $\text{start} = \text{mid}$
  - $\text{start} = \text{mid} + 1$
  - $\text{start} = \text{mid} - 1$

# 死循环的发生

Last Position of Target

nums = [1,1], target = 1

使用  $\text{start} < \text{end}$  无论如何都会出现死循环

## 任意位置 vs 第一个位置 vs 最后一个位置

<http://www.lintcode.com/problem/classical-binary-search/>

<http://www.lintcode.com/problem/first-position-of-target/>

<http://www.lintcode.com/problem/last-position-of-target/>

# Search In a Big Sorted Array

<http://www.lintcode.com/problem/search-in-a-big-sorted-array/>

<http://www.jiuzhang.com/solutions/search-in-a-big-sorted-array/>

# 倍增法 Exponential Backoff

使用到倍增思想的场景：

动态数组 (ArrayList in Java, vector in C++)

网络重试



# Find K Closest Elements

<http://www.lintcode.com/problem/find-k-closest-elements/>

<http://www.jiuzhang.com/solutions/find-k-closest-elements/>

排序数组中找离 target 最接近的 k 个整数

如何提高代码的可读性?

# 第二境界 二分位置 之 OOXX

一般会给你一个数组

让你找数组中第一个/最后一个满足某个条件的位置

OOOOOOO...OXX...XXXXXX

# Maximum Number in Mountain Sequence

<http://www.lintcode.com/problem/maximum-number-in-mountain-sequence/>

<http://www.jiuzhang.com/solutions/maximum-number-in-mountain-sequence/>

在先增后减的序列中找最大值

# Follow Up: 如何实现三分法?

## Trinary Search

问：三分有没有比二分更快？

时间复杂度上会不会更好？

实际运行速度上会不会更快？

# 休息 5 分钟

Take a break

# Find Minimum in Rotated Sorted Array

<http://www.lintcode.com/problem/find-minimum-in-rotated-sorted-array/>

<http://www.jiuzhang.com/solutions/find-minimum-in-rotated-sorted-array/>

**First** position  $\leq$  Last Number

(WRONG: First position  $\leq$  or  $<$  First Number)

# Follow up: 如果有重复的数?

可以证明，无法保证在  $\text{Log}(N)$  的时间复杂度内解决

例子：[1,1,1,1,1...,1] 里藏着一个0

最坏情况下需要把每个位置上的1都看一遍，才能找到最后一个有0的位置

考点：能否想到这个最坏情况的例子

不是写代码！不是写代码！不是写代码！

## 第三境界 一半一半

并无法找到一个条件，形成 XXOO 的模型  
但可以根据判断，保留下有解的那一半或者去掉无解的一半



# Search in Rotated Sorted Array

<http://www.lintcode.com/problem/search-in-rotated-sorted-array/>

<http://www.jiuzhang.com/solutions/search-in-rotated-sorted-array/>

# Find Peak Element

<http://www.lintcode.com/problem/find-peak-element/>

<http://www.jiuzhang.com/solutions/find-peak-element/>

follow up: Find Peak Element II (by 算法强化班)

# 第四境界 二分答案

Binary Search on Answer Range

确定答案范围 + 验证答案大小

# Wood Cut

<https://www.jiuzhang.com/solution/wood-cut/>

<https://www.lintcode.com/problem/wood-cut/>

# 更多的第四境界二分法问题尽在

《九章算法强化班》

<http://www.jiuzhang.com/course/5/>

- Search a 2D Matrix
  - <http://www.lintcode.com/en/problem/search-a-2d-matrix/>
  - <http://www.lintcode.com/en/problem/search-a-2d-matrix-ii/>
    - 不是二分法，但是是常考题
- Search for a Range
  - <http://www.lintcode.com/en/problem/search-for-a-range/>
  - <http://www.lintcode.com/en/problem/total-occurrence-of-target/>
- Smallest Rectangle Enclosing black Pixels
  - <http://www.lintcode.com/problem/smallest-rectangle-enclosing-black-pixels/>
- 以上题目的答案请在 <http://www.jiuzhang.com/solutions> 中搜索

## 课后补充自学内容

- 请在随课教程中查看：<http://www.jiuzhang.com/tutorial/algorithm/>
  - 三步翻转法
  - 二维矩阵找数问题
  - 快速幂算法
  - 辗转相除法
- 点题时间（最近和二分法相关的面试问题）：
  - <http://www.jiuzhang.com/qa/974/>