课程总结

本节内容

1. 现实生活中的算法问题
2. 代码模板
3. 答疑
4. 切题姿势 + 面试四件套
5. 持续练习 + 精深练习

找女朋友问题

- 如果能后悔选之前的：O(n)；
- 如果不能后悔，则用 37% 法则。

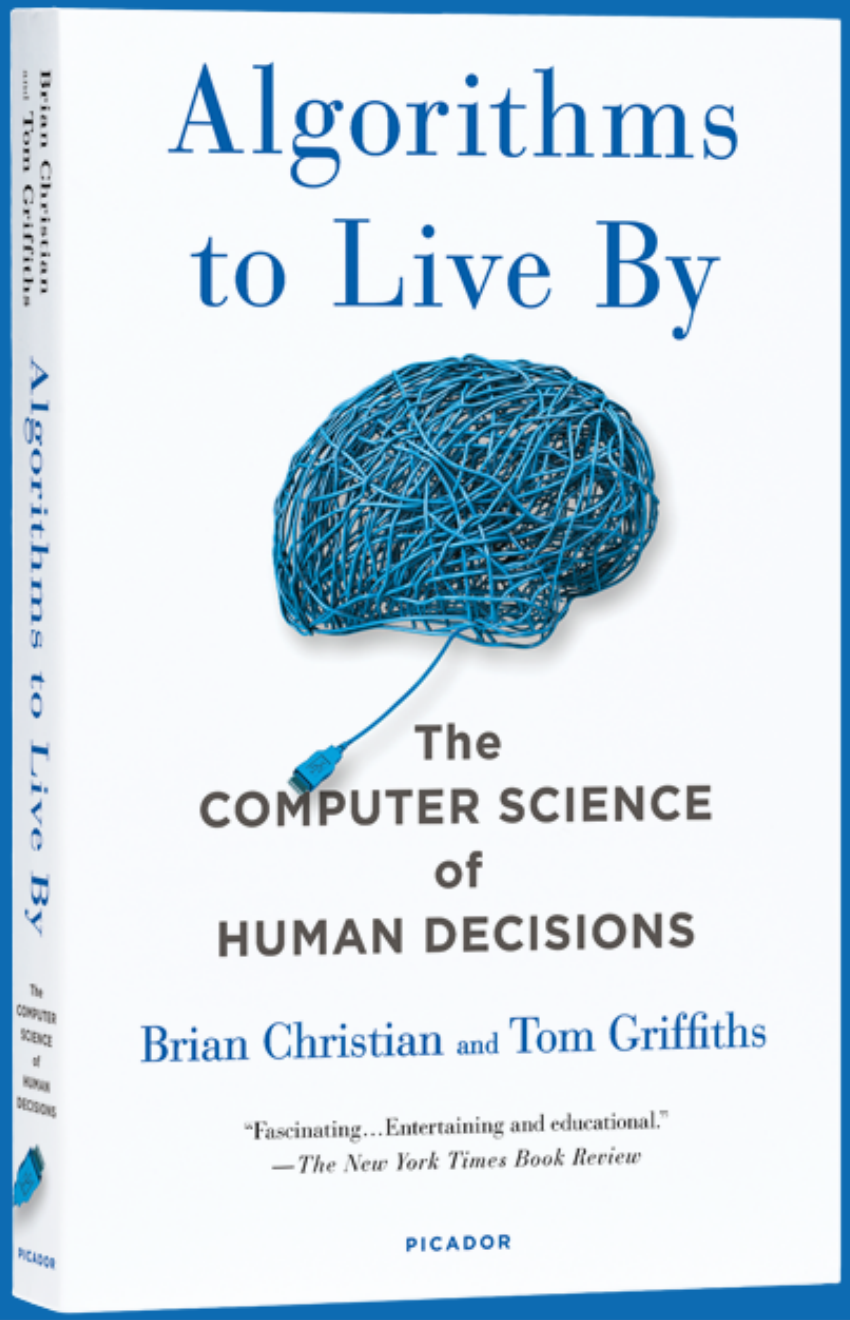
适用范围：

找房子、买东西、换工作等等。

参考阅读：

https://www.jianshu.com/p/9c5c3f839c47

https://blog.csdn.net/s1314_JHC/article/details/78233055



Algorithms to Live By

The COMPUTER SCIENCE of HUMAN DECISIONS

Brian Christian and Tom Griffiths

其他

1. Priority Queue　　一个任务的密度 = 重要程度/完成时间

2. Kelly Formula　　凯利公式

3. Game Theory　　博弈论

代码模板

```
def recursion(level, param1, param2, ...):

  # recursion terminator
  if level > MAX_LEVEL:
    print_result
    return

  # process logic in current level
  process_data(level, data...)

  # drill down
  self.recursion(level + 1, p1, ...)

  # reverse the current level status if needed
  reverse_state(level)
```

DFS代码 - 递归写法

```
visited = set()
def dfs(node, visited):
    visited.add(node)
    # process current node here.
    ...
    for next_node in node.children():
        if not next_node in visited:
            dfs(next_node, visited)
```

```
def BFS(graph, start, end):

    queue = []
    queue.append([start])
    visited.add(start)

    while queue:
        node = queue.pop()
        visited.add(node)

        process(node)
        nodes = generate_related_nodes(node)
        queue.push(nodes)

    # other processing work
    ...
```

```
left, right = 0, len(array) -  1
while left <= right:
    mid = left + (right - left)/2
    if array[mid] == target:
        # find the target!!
        break or return result
    elif array[mid] < target:
        left = mid + 1
    else:
        right = mid -1
```

```
// 状态定义
dp = new int [m + 1 ][n + 1];

// 初始状态
dp[0][0] = x;
dp[0][1] = y;
...

// DP状态的推导
for i = 0; i <= n; ++i {
    for j = 0; j <= m; ++j {
      ...

      d[i][j] = min {dp[i - 1][j], dp[i][j - 1], etc.}
    }
}

return dp[m][n]; // 最优解
```

位运算操作

1. X & 1 == 1 OR == 0 判断奇偶 (X % 2 == 1)

2. X = X & (X-1) => 清零最低位的 1

3. X & -X => 得到最低位的 1

答疑

不同编程语言的语法问题

Python: x, y = 1, 2

Java or C++: x=1; y=2;

Python: x, y = y, x （可以有效实现交换两数）

Java or C++: int tmp = x; x = y; y = tmp;

反转链表 - reverse linked list

```
def reverseList(self, head):
    cur, prev = head, None
    while cur:
        cur.next, prev, cur = prev, cur, cur.next
    return prev
```

链表交换相邻元素


```
def swapPairs(self, head):
    pre, pre.next = self, head
    while pre.next and pre.next.next:
        a = pre.next
        b = a.next
        pre.next, b.next, a.next = b, a, b.next
        pre = a
    return self.next
```

链表交换相邻元素 — Better Version


```
def swapPairs(self, head):
    result = ListNode(0)  # dummy node
    pre, pre.next = result, head
    while pre.next and pre.next.next:
        a = pre.next
        b = a.next
        pre.next, b.next, a.next = b, a, b.next
        pre = a
    return result.next
```

练习和切题

持续练习 + 精深练习（刻意练习）

1. 除了“做熟悉和会做的题目”之外，去刻意练习自己不熟悉的算法和数据结构。

   不要为了切题而切题

2. 做过的题目后续要返回再复习

Top Hits

- Top 100 Liked Questi...
- Top Amazon Ques... 🔒
- Top Facebook Que... 🔒
- Top Google Questi... 🔒
- Top Interview Questi...
- Top LinkedIn Ques... 🔒

Topics

- Array 151
- Dynamic Programming 123
- String 118
- Math 115
- Tree 96
- Hash Table 92
- Depth-first Search 89
- Binary Search 60
- Two Pointers 50
- Breadth-first Search 46
- Greedy 45
- Stack 44
- Backtracking 39
- Design 38
- Linked List 34
- Bit Manipulation 32
- Heap 31
- Sort 21
- Graph 20
- Union Find 19
- Divide and Conquer 17
- Binary Search Tree 14
- Trie 14
- Recursion 11
- Queue 9
- Segment Tree 9
- Random 6
- Binary Indexed Tree 5
- Minimax 5
- Topological Sort 5
- Brainteaser 3
- Geometry 2
- Map 2
- Rejection Sampling 2
- Reservoir Sampling 2
- Memoization 1

面试答题四件套

1. **Clarification**（询问题目细节、边界条件、可能的极端错误情况）
2. **Possible Solution**（所有可能的解法都和面试官沟通一遍）
    - Compare Time & Space Complexity（时间复杂度&空间复杂度）
    - Optimal Solution（最优解）
3. **Coding**（写代码）
4. **Test Cases**（测试用例）

最后：沟通和交流很重要

回到起点

斐波拉契数列（Fibonacci）

拜托，面试别再问我斐波那契数列了！！！

原文链接： https://mp.weixin.qq.com/s/3LR-iVC4zgj0tGhZ780PcQ

姐妹篇：

1.《拜托，面试别再问我TopK了！！！》 *https://mp.weixin.qq.com/s/FFsvWXiaZK96PtUg-mmtEw*

2.《拜托，面试别再让我数1了！！！》 *https://mp.weixin.qq.com/s/A3dLW92SNag8lw7vrQiEHQ*

最后的最后

环境准备

1. Keyboard set-up

2. iTerms + Oh-my-zsh

3. IDE (Pycharm, IntelliJ, Webstorm, GoLand)

   Editors (VS Code, Sublime, Atom, VIM, etc)

切题姿势

结束语

算法和数据结构是内力

重在练习（修行）