

Coding Interview Practice

Suprakash Datta
datta [at] eecs.yorku.ca

Session 1: June 3, 2020
Please do not distribute

The Big Picture

- Software jobs have high salaries, so there is intense competition
- Multi-level selection process, not just at the top companies
- Testing an array of skills (behavioural, problem-solving, coding, testing, explaining ideas...)

Formulating A Plan

To contend you should

- Build your resume systematically, over time
- Open a github account and populate with side projects
- Prepare for behavioural interviews
- Prepare for technical interviews
 - **Coding interview** (computer or whiteboard) - a few hours a week in early years, more in upper years
 - System design questions (paper)

Technical Interviews: Format

- Varies widely
- Popular format:
 - 45 minutes, with interviewer or 90 minutes online, or ...
 - 1-4 questions, often some hard ones
 - whiteboard or computer
 - May need to explain as you go along
- Other formats: many hours long, not monitored

My Involvement

My interest in this:

- I teach Algorithms often and Data Structures occasionally
- This informs my teaching
- I try to keep informed of industry expectations in general, not just recruitment

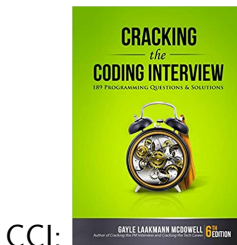
Technical Interviews: Skills

Test knowledge of

- Data Structures, Algorithms, Complexity Analysis
- Coding
- Testing
- Problem solving
 - Initial analysis and design
 - Optimization

Technical Interviews: Essential Resources

- Courses: EECS 2030, EECS 2011, EECS 3101, EECS 4101...
- Books:



- Online practice portal: e.g., Leetcode, Geeksforgeeks,
- Many web resources
- For advanced students: programming contests

Technical Interviews: Topics

- Essential: Arrays, linked lists, heaps, stacks and queues, graphs and trees, problems involving basic math, recursion, sorting and searching, basic properties of binary search trees
- Intermediate and Advanced: Dynamic programming, height balanced binary search trees, tries, string matching algorithms


Note: courses may emphasize reasoning or proving correctness; interview questions usually do not

Technical Interviews: Plan

- It is extremely important to stick to a plan and not get discouraged
- Our plan for this series:
 - Read suggested CCI chapters/sections
 - Try online practice on problems from those chapters
 - Discussion of problems in the interactive sessions

We will move fast; try to keep up as best you can

And now....

- Let's try some problems!
- Our topic for today is: Arrays
- Challenging problems alert 

Problem 1

- Leetcode 217: Given an array of integers, find if the array contains any duplicates.
Your function should return true if any value appears at least twice in the array, and it should return false if every element is distinct.
- Brute force: search for the i -th element in the whole array.
Worst case $\Theta(n^2)$
- Sort and then traverse the array looking for successive repeats.
Worst case $\Theta(n \lg n)$
- How to optimize further?
Can the lookup be $\Theta(1)$ instead of $\Theta(n)$?

Aside: Locating elements in a collection

- Linear Search
- Sorting plus binary search
- Height balanced binary search trees
- Hashing
 - Faster look up time than arrays ($\Theta(1)$ vs. $\Theta(n)$)¹
 - Should be used to store data over arrays wherever possible

¹Disclaimer: amortized complexity

Aside: Hash Tables in Python

- **Sets:** Unordered group of variables (integers, strings, etc.)
- **Dictionaries:** Unordered group of Key/Value pairs
 - **Example:** `dict = {'a': 10, 'b': 20, 'c': 30}`
 - a,b,c are keys and 10,20,30 are values
- Sets cannot contain duplicate elements, dictionaries cannot contain duplicate keys.

Look at documentation for more details and operations with sets/dictionaries:

<https://docs.python.org/3/tutorial/datastructures.html#sets>

<https://docs.python.org/3/tutorial/datastructures.html#dictionaries>

Back to Problem 1

- Leetcode 217: Given an array of integers, find if the array contains any duplicates.

Code

```
1  def containsDuplicate(nums: List[int]) -> bool:  
2      seen = set()  
3      for num in nums:  
4          if num in seen:  
5              return True  
6          else:  
7              seen.add(num)  
8      return False
```

Another Version

- Problem 1: Leetcode 217 Given an array of integers, find if the array contains any duplicates.

Code

```
1 def containsDuplicate(nums: List[int]) -> bool:
2     n = set(nums)
3     if (len(n) == len (nums)):
4         return False
5     else:
6         return True
```

Problem 2

- Leetcode 53: Given an integer array `nums`, find the contiguous subarray (containing at least one number) which has the largest sum and return its sum.
- Brute force: Try every possible interval. Worst case $\Theta(n^3)$
- Slightly better brute force: do not compute each interval separately. Worst case $\Theta(n^2)$
- Can you optimize further?

Problem 2 - continued

- Greedy approach: Pick the locally optimal move at each step, and that will lead to the globally optimal solution
- Iterate through the array, and update the following:
 - 1 **Current (local) maximum sum:** Either add the current element to the current maximum sum, or reset with just the current element.
 - 2 **Global maximum sum:** Either remains the same, or is updated to the value of the local max.
- Let's code it up on Leetcode

Problem 2: A More Compact Solution

- Leetcode 53: Given an integer array `nums`, find the contiguous subarray (containing at least one number) which has the largest sum and return its sum.

Code

```
1 def maxSubArray(nums: List[int]) -> int:
2     max_sum = curr_sum = nums[0]
3
4     for i in range(1, len(nums)):
5         curr_sum = max(nums[i], curr_sum + nums[i])
6         max_sum = max(curr_sum, max_sum)
7
8     return max_sum
```

Problem 3

- Leetcode 121: Say you have an array for which the i -th element is the price of a given stock on day i . If you were only permitted to complete at most one transaction (i.e., buy one and sell one share of the stock), design an algorithm to find the maximum profit.

Note that you cannot sell a stock before you buy one.

- Choosing the global minimum and maximum does not work
- Brute force: Try every possible price as a buy and iterate over later prices to find the best price to sell. Worst case $\Theta(n^2)$
- Is there a faster solution?

Problem 3: Approach

- Leetcode 121: Say you have an array for which the i -th element is the price of a given stock on day i . If you were only permitted to complete at most one transaction (i.e., buy one and sell one share of the stock), design an algorithm to find the maximum profit.

Note that you cannot sell a stock before you buy one.

- What can we keep track of, so that we can solve the problem in one sweep of the array?

Problem 3: A Solution

- Leetcode 121: Say you have an array for which the i th element is the price of a given stock on day i

Code

```

1  def maxProfit(prices: List[int]) -> int:
2      n = len(prices)
3      if (n==0):
4          return 0
5      minSoFar = prices[0]
6      Profit = 0
7      for i in range(1,n):
8          Profit = max(prices[i]-minSoFar, Profit)
9          minSoFar = min(minSoFar, prices[i])
10
11     return Profit

```

What next?

- Select topics to read up on
- Practice, practice, practice
- My recommendations follow

Recommended Problems

- CCI: 1.3, 1.4 (both easy), 1.9 (Tricky, clever solution given)
- Leetcode:
 - **Easy:** Valid Anagram, Two Sum, First Unique Character in a String
 - **Slightly Harder:** Merge Sorted Array, Group Anagrams, Add Strings
 - **Optional:** 3Sum, Product of Array Except Self, Merge Intervals, Search in Rotated Sorted Array

Note: Some problems may be difficult at first, even for more advanced students, but it is important to at least look over their solutions in order to gain understanding of common strategies that are used in frequently asked interview questions.

Next topic: More Arrays, Linked lists, Stacks, Queues

Questions?