



# LeetCode Bootcamp

Presented By: Spriha Jha

# Session Outline

- 01.** LeetCode platform overview
- 02.** Tech Interview Resources
- 03.** Bootcamp Timeline
- 04.** Introduction to List, Arrays & Sorting
- 05.** Problem Sets
- 06.** Debrief & Q/A

PART 01

---

# LeetCode Platform Overview

Free Tier

Problem  
Statement

LeetCode Explore Problems Interview Contest Discuss Store Premium

Description Solution Discuss (999+) Submissions Python3 Autocomplete

### 1. Two Sum

Easy 38402 1228 Add to List Share

Given an array of integers `nums` and an integer `target`, return *indices of the two numbers such that they add up to `target`*.

You may assume that each input would have **exactly one solution**, and you may not use the same element twice.

You can return the answer in any order.

**Example 1:**

**Input:** `nums = [2,7,11,15], target = 9`  
**Output:** `[0,1]`  
**Explanation:** Because `nums[0] + nums[1] == 9`, we return `[0, 1]`.

**Example 2:**

**Input:** `nums = [3,2,4], target = 6`  
**Output:** `[1,2]`

**Example 3:**

**Input:** `nums = [3,3], target = 6`  
**Output:** `[0,1]`

**Constraints:**

```
1 class Solution:
2     def twoSum(self, nums: List[int], target: int) -> List[int]:
3
```

Problems Pick One < Prev 1/2430 Next > Console Contribute i Run Code ^ Submit

Add your  
code here!

Identifying  
edge-cases



LeetCode Explore Problems Interview Contest Discuss Store

Example 3:

Input: nums = [3,3], target = 6  
Output: [0,1]

Constraints:

- $2 \leq \text{nums.length} \leq 10^4$
- $-10^9 \leq \text{nums}[i] \leq 10^9$
- $-10^9 \leq \text{target} \leq 10^9$
- Only one valid answer exists.

Follow-up: Can you come up with an algorithm that is less than  $O(n^2)$  time complexity?

Accepted 7,918,980 Submissions 16,142,210

Seen this question in a real interview before?

Companies

Related Topics

Similar Questions

Show Hint 1

Show Hint 2

Show Hint 3

```
1 class Solution:
2     def twoSum(self, nums: List[int], target: int) -> List[int]:
3
```

Console - Contribute

PART 02

---

# Tech Interview Resources

Grind75 | CodeSignal | GitHub

# Tech Interview Handbook

GRIND 75

by the author of Blind 75

Star

79,375

[Tech Interview Handbook](#)
[Algorithms Cheatsheet](#)
[Changelog](#)
[FAQ](#)
[About](#)

Indicate your preferences and we will recommend the best LeetCode questions for you to practice.

**SCHEDULE**

8 weeks

8 hours per week

**DIFFICULTY**

☒ Easy

☒ Medium

☒ Hard

**TOPICS**

All topics selected [\(Change\)](#)

## Grind 75 questions

Customize LeetCode study plans according to your needs. You are recommended to work on the questions in order. [Find out why.](#)

**Questions Summary**

**TIME NEEDED**

How long doing these questions will take for an average person. It's a conservative estimate where it is assumed that roughly the same amount of time will be needed to check the answer for each question.

**64 hours**

[Fits into your schedule.](#)

**DIFFICULTY**

Questions grouped by difficulty

Easy: 24 Medium: 42 Hard: 9

**TOPICS**

Questions grouped by topics

Array: 11 Binary: 1

Binary Search: 5

Binary Search Tree: 3 Binary Tree: 9

Dynamic Programming: 5 Graph: 10

Hash Table: 1 Heap: 4

Linked List: 5 Matrix: 1

Recursion: 3 Stack: 7 String: 8

Trie: 2

Need study resources? Check out [Tech Interview Handbook's algorithm study cheatsheets.](#)

COMPLETED 0 / 75

🔧 You can now bookmark the page to save your preferences! We also recently changed the questions presentation settings. Refer to the [changelog](#) for more details.

[Based on Preferences](#)
[Order by Difficulty](#)
[Group by Weeks](#)
[Show topics](#)

**Stop grinding LeetCode. Study with a plan**

Developed by Google engineers, AlgoMonster is the fastest way to get a Software Engineering job. Join today for a 70% discount!

**Master Coding Interviews Effectively**

Grokking the Coding Interview teaches you techniques and question patterns to be good at coding interviews. Get your lifetime access now!

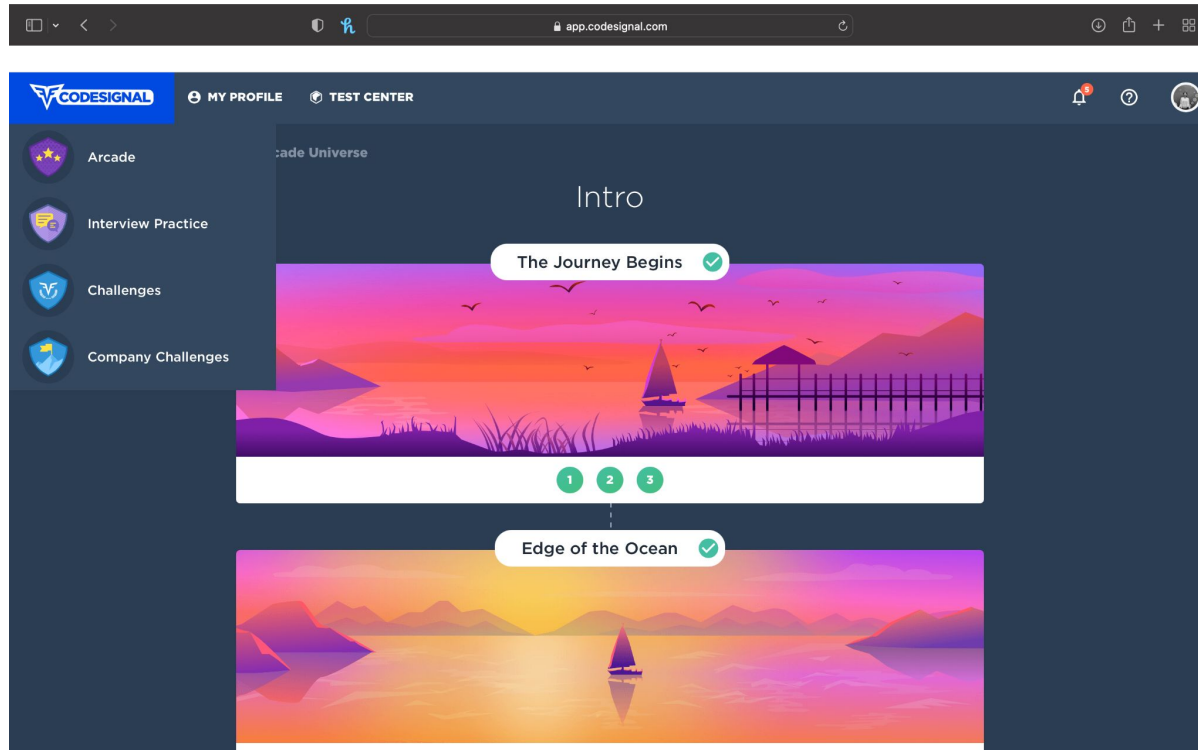
**Craft the Perfect Resume for FAANG**

Save time crafting your resume with FAANG Tech Leads' FAANG: quality resume templates and examples which have helped many Software Engineers get interviews at top Bay Area companies. Grab them now for a whopping 70% off!

**Free curated interview preparation materials**

Tech Interview Handbook goes straight to the point and tells you the minimum you need to know to excel in your technical interviews, and it's free!

# CodeSignal



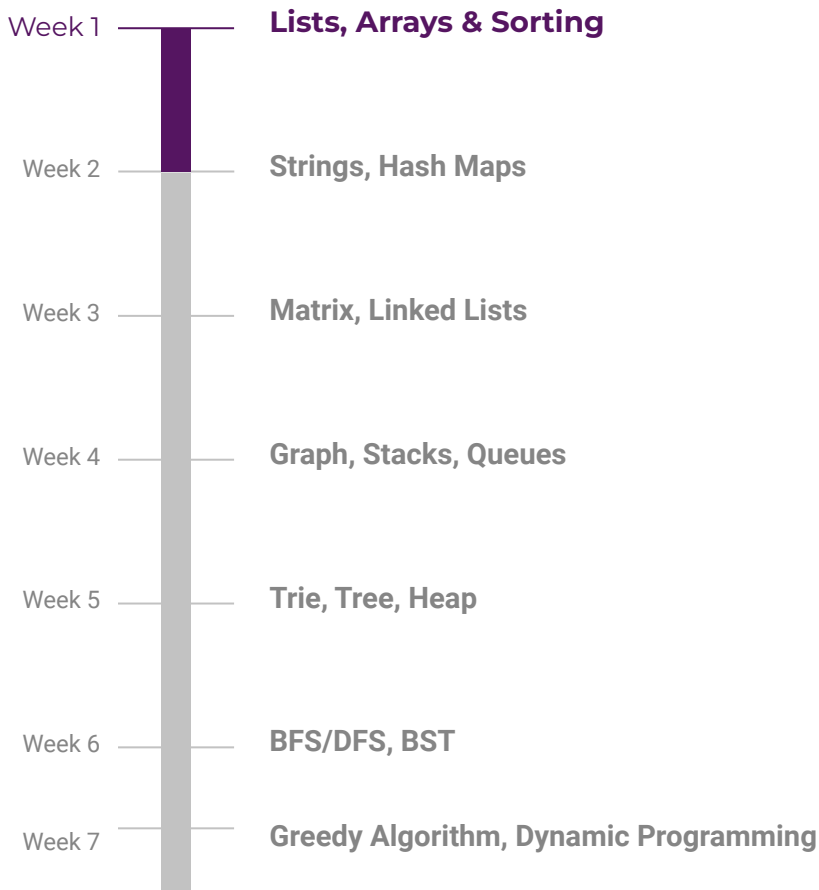


PART 03

---

# Bootcamp Course Structure

# Bootcamp Timeline



# Project Brief

Full-stack (working on both the frontend and backend) web application using an exciting data set.

## Outcome:

1. How to work with APIs?
2. Write frontend code that interacts with APIs.
3. Document the entire project from the design phase to publishing it on GitHub.
4. Web technologies, including HTML, CSS, and JavaScript for beginners and React.js and Next.js for advanced.

# Project Brief

1. Select an exciting data set to build their web applications. Here are some datasets to choose from:

- <http://www.omdbapi.com/> (Movie Title dataset)
- <https://catalog.data.gov/dataset/air-quality> (Air quality in NY)
- <https://catalog.data.gov/dataset/motor-vehicle-collisions-crashes> (Car Crashes in NY)

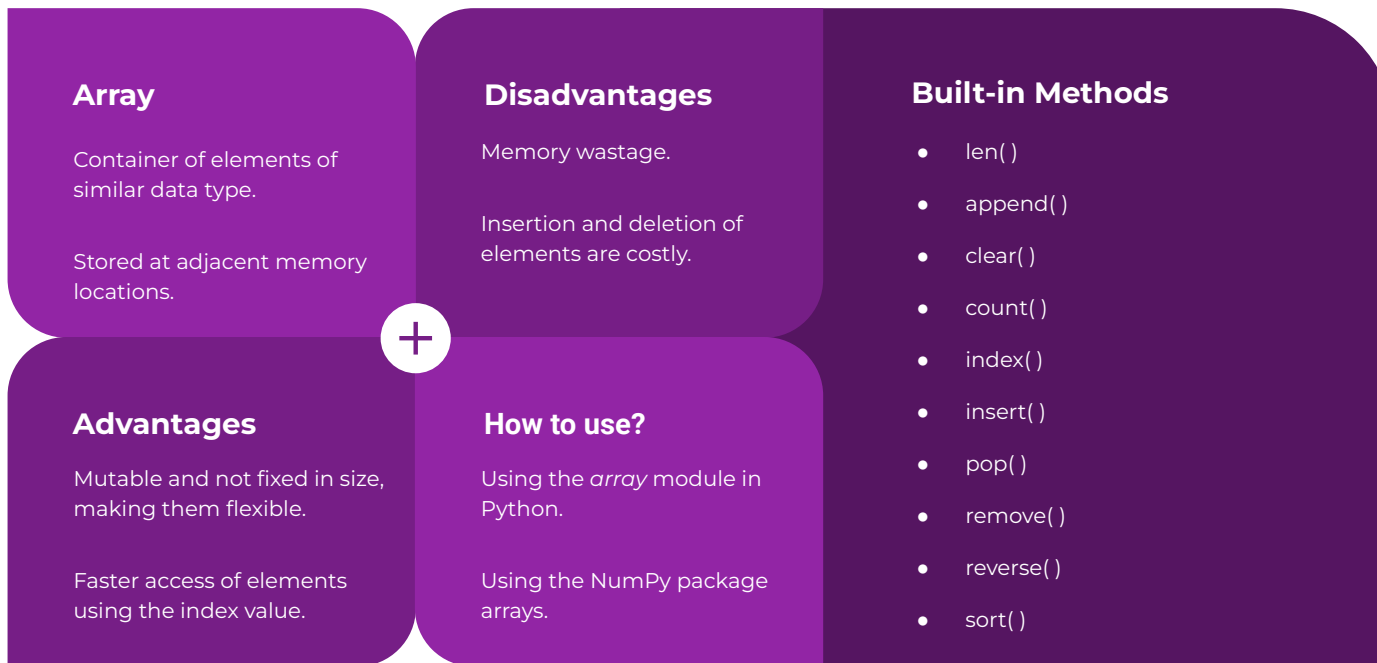
2. User Interface design and developing a web application demo. Some possible projects from the datasets would be:

- Movie Title dataset: A movie renting platform with a shopping cart similar to Amazon.
- Air quality in NY: An interactive air quality map across New York with filters and search features.
- Car Crashes in NY: An interactive map to show the trends of car crashes across New York like a time series.

PART 04

---

# Introduction to Lists, Arrays & Sorting



PART 05

---

# Problem Sets

# Steps to approach the question:

## Understand the problem

Take time to carefully read through the problem from start to finish is critical in finding the correct and complete solution to the problem in hand.

## Code your solution

Map out your solution before you write any code. Avoid too much time trying to find the perfect solution. Validate your solution early and often.

## Manage your time

Don't forget, you have multiple questions to complete within a said time. Make sure you allocate enough time to carefully consider all problems.



# Problem 1: Best time to Buy and Sell Stock

The screenshot shows the LeetCode interface for problem 121, "Best Time to Buy and Sell Stock". The problem is categorized as "Easy" with 20,429 likes and 656 dislikes. The description states: "You are given an array `prices` where `prices[i]` is the price of a given stock on the  $i^{\text{th}}$  day. You want to maximize your profit by choosing a **single day** to buy one stock and choosing a **different day in the future** to sell that stock. Return the *maximum profit* you can achieve from this transaction. If you cannot achieve any profit, return `0`."

**Example 1:**  
Input: `prices = [7,1,5,3,6,4]`  
Output: `5`  
Explanation: Buy on day 2 (price = 1) and sell on day 5 (price = 6), profit = 6-1 = 5.  
Note that buying on day 2 and selling on day 1 is not allowed because you must buy before you sell.

**Example 2:**  
Input: `prices = [7,6,4,3,1]`  
Output: `0`  
Explanation: In this case, no transactions are done and the max profit = 0.

**Constraints:**  
• `1 <= prices.length <= 105`

The right side of the screenshot shows a code editor with the following Python code:

```
1 class Solution:
2     def maxProfit(self, prices: List[int]) -> int:
3
```

At the bottom, there are navigation buttons: "Problems", "Pick One", "< Prev", "121/2430", "Next >", "Console", "Contribute i", "Run Code ^", and "Submit".

## Approach 1: Brute Force

class Solution:

```
def maxProfit(self, prices: List[int]) -> int:
```

```
    max_profit = 0
```

```
    for i in range(len(prices) - 1):
```

```
        for j in range(i + 1, len(prices)):
```

```
            profit = prices[j] - prices[i]
```

```
            if profit > max_profit:
```

```
                max_profit = profit
```

```
    return max_profit
```

**Time complexity:**  $O(n^2)$ , loop runs  $n(n-1)/2$  times

**Space complexity:**  $O(1)$ , only two variables *max\_profit* and *profit*

## Approach 2: Single Pass

class Solution:

```
def maxProfit(self, prices: List[int]) -> int:
```

```
    min_price = prices[0]
```

```
    max_profit = 0
```

```
    for price in prices:
```

```
        min_price = min(min_price, price)
```

```
        max_profit = max(price - min_price, max_profit)
```

```
    return max_profit
```

**Time complexity:**  $O(n)$ , only single pass is needed

**Space complexity:**  $O(1)$ , only two variables *max\_profit* and *min\_price*

# Problem 2: Sort Colors

LeetCode

Explore Problems Interview 75% Contest Discuss Store

Description

Solution

Discuss (999+)

Submissions

75. Sort Colors

Medium 12392 464 Add to List Share

Given an array `nums` with `n` objects colored red, white, or blue, sort them in-place so that objects of the same color are adjacent, with the colors in the order red, white, and blue.

We will use the integers `0`, `1`, and `2` to represent the color red, white, and blue, respectively.

You must solve this problem without using the library's sort function.

**Example 1:**

**Input:** `nums = [2,0,2,1,1,0]`  
**Output:** `[0,0,1,1,2,2]`

**Example 2:**

**Input:** `nums = [2,0,1]`  
**Output:** `[0,1,2]`

**Constraints:**

- `n == nums.length`
- `1 <= n <= 300`
- `nums[i]` is either `0`, `1`, or `2`.

**Follow up:** Could you come up with a one-pass algorithm using only constant extra space?

Python3

Autocomplete

```
1 class Solution:
2     def sortColors(self, nums: List[int]) -> None:
3         """
4         Do not return anything, modify nums in-place instead.
5         """
6         
```

Problems

Pick One

< Prev 75/2430 Next >

Console Contribute i

Run Code ^

Submit

## Approach 1: Single Pass

class Solution:

```
def sortColors(self, nums: List[int]) -> None:
```

```
    p0 = curr = 0
```

```
    p2 = len(nums) - 1
```

```
    while curr <= p2:
```

```
        if nums[curr] == 0:
```

```
            nums[p0], nums[curr] = nums[curr], nums[p0]
```

```
            p0 += 1
```

```
            curr += 1
```

```
        elif nums[curr] == 2:
```

```
            nums[curr], nums[p2] = nums[p2], nums[curr]
```

```
            p2 -= 1
```

```
        else:
```

```
            curr += 1
```

**Time complexity:**  $O(n)$ , loop runs one pass along the length

**Space complexity:**  $O(1)$ , constant space

# Problem 3: Group Anagrams

The screenshot shows the LeetCode interface for problem 49, 'Group Anagrams'. The page includes tabs for Description, Editorial, Solutions (7.4K), and Submissions. The problem is rated 'Medium' with 14.5K likes and 423 comments. It is tagged with 'Amazon', 'Yandex', and 'Bloomberg'. The description explains that an anagram is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once. It provides three examples: Example 1 with input ["eat", "tea", "tan", "ate", "nat", "bat"] and output [["bat"], ["nat", "tan"], ["ate", "eat", "tea"]]; Example 2 with input [""] and output [[]]; and Example 3 with input ["a"] and output [["a"]]. Constraints are listed: 1 ≤ strs.length ≤ 10<sup>4</sup>, 0 ≤ strs[i].length ≤ 100, and strs[i] consists of lowercase English letters. On the right, a Python3 code editor shows a class Solution with a method groupAnagrams that takes a list of strings and returns a list of lists of strings. The interface also features a 'Problem List' button, a 'Console' button, and 'Run' and 'Submit' buttons at the bottom right.

LeetCode

< Problem List >

Description Editorial Solutions (7.4K) Submissions

Python3 Auto

### 49. Group Anagrams

Medium 14.5K 423

Amazon Yandex Bloomberg

Given an array of strings `strs`, group **the anagrams** together. You can return the answer in **any order**.

An **Anagram** is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once.

**Example 1:**

Input: `strs = ["eat", "tea", "tan", "ate", "nat", "bat"]`  
Output: `[["bat"], ["nat", "tan"], ["ate", "eat", "tea"]]`

**Example 2:**

Input: `strs = [""]`  
Output: `[[""]]`

**Example 3:**

Input: `strs = ["a"]`  
Output: `[["a"]]`

**Constraints:**

- `1 ≤ strs.length ≤ 104`
- `0 ≤ strs[i].length ≤ 100`
- `strs[i]` consists of lowercase English letters.

```
1 class Solution:
2     def groupAnagrams(self, strs: List[str]) -> List[List[str]]:
```

Console

Run Submit

## Approach 1: Categorize by Sorted String

```
class Solution(object):  
    def groupAnagrams(self, strs):  
        ans = collections.defaultdict(list)  
        for s in strs:  
            ans[tuple(sorted(s))].append(s)  
        return ans.values()
```

**Time complexity:**  $O(NK\log K)$ ,  $N$  is the length of `strs`, and  $K$  is the maximum length of a string in `strs`

**Space complexity:**  $O(NK)$ , constant space  $NK$

## Approach 2: Categorize by Count

class Solution:

```
def groupAnagrams(strs):  
    ans = collections.defaultdict(list)  
    for s in strs:  
        count = [0] * 26  
        for c in s:  
            count[ord(c) - ord('a')] += 1  
        ans[tuple(count)].append(s)  
    return ans.values()
```

**Time complexity:**  $O(NK)$ ,  $N$  is the length of `strs`, and  $K$  is the maximum length of a string in `strs`

**Space complexity:**  $O(NK)$ , constant space  $NK$



PART 06

---

Q/A

# Slack Invite

[Join Slack Workspace!](#)

# Problem Assignments

- 01.** Running Sum of 1d Array (Easy)
- 02.** Long Pressed Name (Easy)
- 03.** Contains Duplicate (Easy)
- 04.** Video Stitching (Medium)
- 05.** Maximum Product Subarray (Medium)
- 06.** Container With Most Water (Medium)
- 07.** Sliding Window Maximum (Hard)



# Thank you!

Upcoming: Strings & Hash Maps