



# Leetcamp Week 4

Zixuan Jia

# Q. 5 Longest Palindromic Substring

## 5. Longest Palindromic Substring

Medium

👍 8413

💬 590

♡ Add to List

🔗 Share

Given a string `s`, return *the longest palindromic substring* in `s`.

### Example 1:

**Input:** `s = "babad"`

**Output:** `"bab"`

**Note:** `"aba"` is also a valid answer.

### Example 2:

**Input:** `s = "cbbd"`

**Output:** `"bb"`

### Example 3:

**Input:** `s = "a"`

**Output:** `"a"`

### Example 4:

**Input:** `s = "ac"`

**Output:** `"a"`



# Solution 1: Brute Force

- Picking all possible starting and ending positions for a substring
- Check if it is a substring
- $O(n^3)$
- Terrible solution



## Solution 2: Expand from Middle

- Find the midpoint and expand from it, check if it valid
- $O(n^2)$



# Better Solution?

- Yes but not necessary
- Manacher's Algorithm
- [https://en.wikipedia.org/wiki/Longest\\_palindromic\\_substring#Manacher's\\_algorithm](https://en.wikipedia.org/wiki/Longest_palindromic_substring#Manacher's_algorithm)

# Q53. Maximum Subarray

## 53. Maximum Subarray

Easy 9411 445 Add to List Share

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

**Follow up:** If you have figured out the  $O(n)$  solution, try coding another solution using the **divide and conquer** approach, which is more subtle.

### Example 1:

**Input:** `nums = [-2,1,-3,4,-1,2,1,-5,4]`  
**Output:** 6  
**Explanation:** [4,-1,2,1] has the largest sum = 6.

### Example 2:

**Input:** `nums = [1]`  
**Output:** 1

### Example 3:

**Input:** `nums = [0]`  
**Output:** 0

### Example 4:

**Input:** `nums = [-1]`  
**Output:** -1

### Example 5:

**Input:** `nums = [-2147483647]`  
**Output:** -2147483647



# Solution 1: Brute Force

- Double for loop go through the whole array
- Check the sum between 2 pointers and save the answer and find the maximum



## Solution 2: Sliding window

- Similar to 2 pointers
- Check the sum within the sub-array
- Slide the start or end



# #125 Valid Palindrome

## 125. Valid Palindrome

Easy

👍 1500

💬 3254

❤️ Add to List

📄 Share

Given a string, determine if it is a palindrome, considering only alphanumeric characters and ignoring cases.

**Note:** For the purpose of this problem, we define empty string as valid palindrome.

### Example 1:

Input: "A man, a plan, a canal: Panama"

Output: true

### Example 2:

Input: "race a car"

Output: false



# Two Pointers

- Start from left and right of the string and check each character
- If it is a space/punctuation, move to the next one
- Only compare alphanumeric values

# #56 Merge Intervals

## 56. Merge Intervals

Medium

👍 5377

🔖 324

💖 Add to List

🔗 Share

Given a collection of intervals, merge all overlapping intervals.

### Example 1:

Input: intervals = [[1,3],[2,6],[8,10],[15,18]]

Output: [[1,6],[8,10],[15,18]]

Explanation: Since intervals [1,3] and [2,6] overlaps, merge them into [1,6].

### Example 2:

Input: intervals = [[1,4],[4,5]]


Output: [[1,5]]

Explanation: Intervals [1,4] and [4,5] are considered overlapping.

**NOTE:** input types have been changed on April 15, 2019. Please reset to default code definition to get new method signature.

### Constraints:

- `intervals[i][0] <= intervals[i][1]`

- 
- Sort the list
  - If the first number (start) of the next interval is smaller than then second number (end) of the previous interval, compare the second number of the next interval with the second number of the previous interval and assign the maximum of the two to the second number of the previous interval
  - Else append the interval to the new list