Tutorial 1: Introducing the Problem Statement

01

What is Leetcode all about?

Good Luck :)

Basic Structure

Problem Title

Problem Statement.

Typically, this will be a description of the problem being solved.

This may include some sort of story being told.

Constraints and structure of how input is given.

Expected Input

A sample test case is provided for you to test your program on.

Expected Output

The solution for this expected program.

Why solve informatics problems?

Benefits

- Improves your pattern recognition
- Improves your familiarity with programming structure and languages
- Provides a different perspective to situations
- Keeps your mind active

Cons

- Takes up a lot of time
- May get tendonitis in fingers and bad posture

Solving Problems

Problem Solving

You have to write some program that takes in the expected input and produces the expected output.

How to practice

Solve more problems! The more problems you solve, the better you'll get. Supplement that with regular review of past problems you've solved, and you're golden.

Where

Leetcode if you're mainly focused on interview prep. Otherwise, Codeforces/AtCoder are my favourite platforms.

02

Example Problems

Good Luck!

Valid Parentheses

https://leetcode.com/problems/valid-parentheses/description/

Given a string s containing just the characters '(', ')', '{', '}', '[' and ']', determine if the input string is valid.

An input string is valid if:

1. Open brackets must be closed by the same type of brackets.
2. Open brackets must be closed in the correct order.
3. Every close bracket has a corresponding open bracket of the same type.

Valid Parentheses

https://leetcode.com/problems/valid-parentheses/description/

Questions to ask yourself as you read through the problem

- What does it mean when an input string is valid?
- What constraints does a valid input have?
- What are some things that a valid input must not have?
- What are some things that a valid input must have?

Integer to Roman

https://leetcode.com/problems/integer-to-roman/description/

Roman numerals are formed by appending the conversions of decimal place values from highest to lowest. Converting a decimal place value into a Roman numeral has the following rules:

- If the value does not start with 4 or 9, select the symbol of the maximal value that can be subtracted from the input, append that symbol to the result, subtract its value, and convert the remainder to a Roman numeral.
- If the value starts with 4 or 9 use the **subtractive form** representing one symbol subtracted from the following symbol, for example, 4 is 1 (`I`) less than 5 (`V`): `IV` and 9 is 1 (`I`) less than 10 (`X`): `IX`. Only the following subtractive forms are used: 4 (`IV`), 9 (`IX`), 40 (`XL`), 90 (`XC`), 400 (`CD`) and 900 (`CM`).
- Only powers of 10 (`I`, `X`, `C`, `M`) can be appended consecutively at most 3 times to represent multiples of 10. You cannot append 5 (`V`), 50 (`L`), or 500 (`D`) multiple times. If you need to append a symbol 4 times use the **subtractive form**.

Given an integer, convert it to a Roman numeral.

Integer to Roman

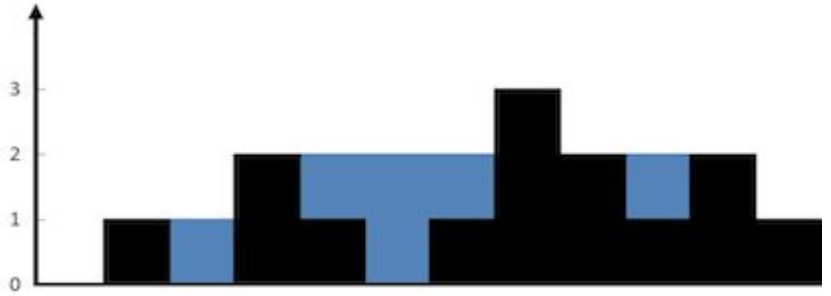
Questions to ask yourself as you read through the problem

- How would we solve some of the examples?
- What is the largest number we are given?
- When would you use each roman numeral? I.e., when would we use ‘I’, ‘X’, ‘M’, etc..
- Is a decimal ten number to its roman-numeral version deterministic? I.e., is there only one valid roman numeral for every integer?

Trapping Rain Water

https://leetcode.com/problems/trapping-rain-water/description

Given n non-negative integers representing an elevation map where the width of each bar is 1, compute how much water it can trap after raining.



Trapping Rain Water

Questions to ask yourself as you read through the problem

- What does it mean when a block traps rain water?
- When would a block trap some rain water?

Container With Most Water

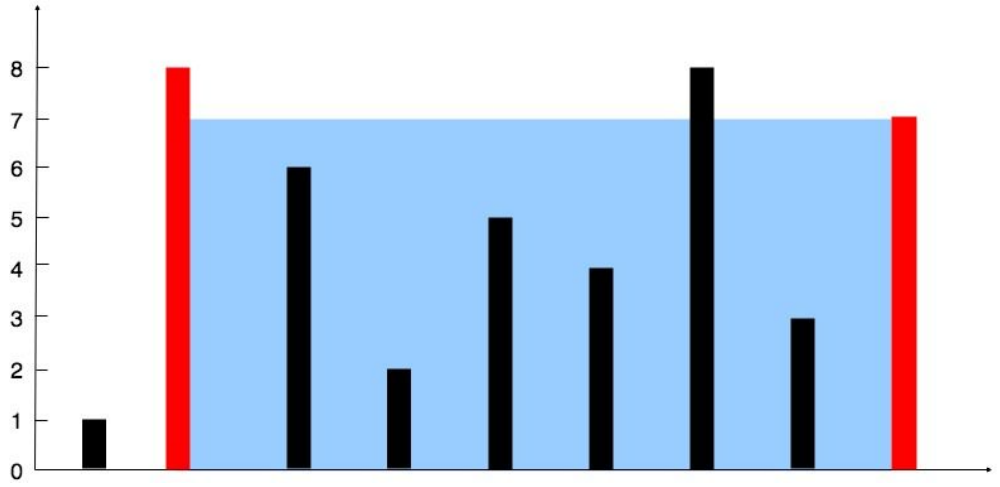
https://leetcode.com/problems/container-with-most-water/description/

You are given an integer array `height` of length `n`. There are `n` vertical lines drawn such that the two endpoints of the i^{th} line are `(i, 0)` and `(i, height[i])`.

Find two lines that together with the x-axis form a container, such that the container contains the most water.

Return *the maximum amount of water a container can store*.

Notice that you may not slant the container.



Container With Most Water

Questions to ask yourself as you read through the problem

- How much water do two walls store?
- How does this change as you go from one wall to the next?

This is quite the confusing problem! It's technically a Leetcode "medium", but it's the hardest problem you'll solve today. Let it be a lesson that problem descriptions aren't always perfect.