

## ISM 6225

**Distributed Information Systems**

Assignment 2 – Computational Problem Solving

**Primary objective**: Develop familiarity with algorithms, data structures (arrays and dictionaries)

**Secondary objective**: Develop comfort with using GitHub and the IDE

# Introduction

Conversations with recent graduates suggests that employers expect most students to be comfortable with standard questions on problem solving, algorithms and data structures. Students have also shared that their prior experience often did not give them adequate introduction to algorithms, data structures and problem solving to be competitive in the market.

This assignment aims to give students the opportunity to familiarize themselves with problem solving, while simultaneously using basic sorting and searching algorithms to develop familiarity with algorithms and using simple data structures such as arrays, strings, dictionaries, and hash sets to develop familiarity with the various data structures.

This is an individual assignment. To accomplish this, students are expected to use the GitHub source control to develop their solution.

# Activity

All the methods are defined in Program.cs. Complete their definitions to successfully run the program.

You are free to modify the values of the arrays and variables in the main method.

## NOTE 1: Don’t edit the method’s declaration, i.e., you cannot change the parameter type or the return type.

**Note 2: For full credit, your solution should meet the time and/ or space complexity constraints as well as any data structure/ algorithm recommendations specified.**

Submission

Push the code to GitHub and submit the URL. Also, get the output from a sample run that shows the use of all required methods, take the screenshot, and upload/push it to GitHub. Submit the self-reflection as a comment in Canvas.

# Grading scheme:

Each question carries 1.875 points. You will be graded on the following aspects: Logic (including appropriate organization of logic into methods) :1.375

Handling all reasonable corner cases : 0.25 Self-reflection (learning from the question, and recommendations) : 0.1 Code comments : 0.15

# GitHub Link

## Use the Program.CS File from this [GitHub Repository](https://github.com/ISM6225/ISM6225_Fall_2023_Assignment_2/tree/main/ISM6225_Fall_2023_Assignment_2).

**Question 1:**

You are given an inclusive range [lower, upper] and a sorted unique integer array nums, where all elements are within the inclusive range. A number x is considered missing if x is in the range [lower, upper] and x is not in nums. Return the shortest sorted list of ranges that exactly covers all the missing numbers. That is, no element of nums is included in any of the ranges, and each missing number is covered by one of the ranges.

Example 1:

Input: nums = [0,1,3,50,75], lower = 0, upper = 99

Output: [[2,2],[4,49],[51,74],[76,99]]

Explanation: The ranges are:

[2,2]

[4,49]

[51,74]

[76,99]

Example 2:

Input: nums = [-1], lower = -1, upper = -1

Output: []

Explanation: There are no missing ranges since there are no missing numbers.

Constraints:

-109 <= lower <= upper <= 109

0 <= nums.length <= 100

lower <= nums[i] <= upper

All the values of nums are unique.

Time complexity: O(n), space complexity:O(1)

**Question 2:**

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

Open brackets must be closed by the same type of brackets.

Open brackets must be closed in the correct order.

Every close bracket has a corresponding open bracket of the same type.

Example 1:

Input: s = "()"

Output: true

Example 2:

Input: s = "()[]{}"

Output: true

Example 3:

Input: s = "(]"

Output: false

Constraints:

1 <= s.length <= 104

s consists of parentheses only '()[]{}'.

Time complexity:O(n^2), space complexity:O(1)

**Question 3:**

You are given an array prices where prices[i] is the price of a given stock on the ith 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

0 <= prices[i] <= 104

Time complexity: O(n), space complexity:O(1)

**Question 4:**

Given a string num which represents an integer, return true if num is a strobogrammatic number. A strobogrammatic number is a number that looks the same when rotated 180 degrees (looked at upside down).

Example 1:

Input: num = "69"

Output: true

Example 2:

Input: num = "88"

Output: true

Example 3:

Input: num = "962"

Output: false

Constraints:

1 <= num.length <= 50

num consists of only digits.

num does not contain any leading zeros except for zero itself.

Time complexity:O(n), space complexity:O(1)

**Question 5:**

Given an array of integers nums, return the number of good pairs.A pair (i, j) is called good if nums[i] == nums[j] and i < j.

Example 1:

Input: nums = [1,2,3,1,1,3]

Output: 4

Explanation: There are 4 good pairs (0,3), (0,4), (3,4), (2,5) 0-indexed.

Example 2:

Input: nums = [1,1,1,1]

Output: 6

Explanation: Each pair in the array are good.

Example 3:

Input: nums = [1,2,3]

Output: 0

Constraints:

1 <= nums.length <= 100

1 <= nums[i] <= 100

Time complexity:O(n), space complexity:O(n)

**Question 6:**

Given an integer array nums, return the third distinct maximum number in this array. If the third maximum does not exist, return the maximum number.

Example 1:

Input: nums = [3,2,1]

Output: 1

Explanation:

The first distinct maximum is 3.

The second distinct maximum is 2.

The third distinct maximum is 1.

Example 2:

Input: nums = [1,2]

Output: 2

Explanation:

The first distinct maximum is 2.

The second distinct maximum is 1.

The third distinct maximum does not exist, so the maximum (2) is returned instead.

Example 3:

Input: nums = [2,2,3,1]

Output: 1

Explanation:

The first distinct maximum is 3.

The second distinct maximum is 2 (both 2's are counted together since they have the same value).

The third distinct maximum is 1.

Constraints:

1 <= nums.length <= 104

-231 <= nums[i] <= 231 - 1

Time complexity:O(nlogn), space complexity:O(n)

**Question 7:**

You are playing a Flip Game with your friend. You are given a string currentState that contains only '+' and '-'. You and your friend take turns to flip two consecutive "++" into "--". The game ends when a person can no longer make a move, and therefore the other person will be the winner.Return all possible states of the string currentState after one valid move. You may return the answer in any order. If there is no valid move, return an empty list [].

Example 1:

Input: currentState = "++++"

Output: ["--++","+--+","++--"]

Example 2:

Input: currentState = "+"

Output: []

Constraints:

1 <= currentState.length <= 500

currentState[i] is either '+' or '-'.

Timecomplexity:O(n), Space complexity:O(n)

**Question 8:**

Given a string s, remove the vowels 'a', 'e', 'i', 'o', and 'u' from it, and return the new string.

Example 1:

Input: s = "leetcodeisacommunityforcoders"

Output: "ltcdscmmntyfrcdrs"

Example 2:

Input: s = "aeiou"

Output: ""

Timecomplexity:O(n), Space complexity:O(n)