Assignment - II

Backtracking, Stack and Queue, Merge/Quick/Radix, Binary Tree, Binary Heap/ Priority Queue

Note: You have to submit a .cpp file for every question. The examples of input and output are for understanding only. Your code should be generalized i.e., work on any related test case.
The deadline for the assignment is a day before the **Mid-II Exam.**
No late submissions will be entertained.

In case of Plagiarism, Straight Zero marks will be assigned.

<u>Backtracking</u>: 20 Marks

Imagine you're tasked with repairing a broken calculator. The damage is strange - the calculator only works when presented with a specific sequence of numbers. Each number is the sum of the two previous ones you entered. The challenge is, the starting numbers are missing!

Your job is to rewrite the calculator's logic. Given a string of digits that represent the button presses on the broken calculator, write a program to determine if it could have produced a valid calculation sequence. Remember, the sequence must have at least three numbers, and the calculator can't handle leading zeros (except for the single digit zero itself). By analyzing the button presses, can you tell if the calculator would have functioned correctly?

Example 1:
Input: "112358"
Output: true
Explanation:
The digits can form an additive sequence: 1, 1, 2, 3, 5, 8.
1 + 1 = 2, 1 + 2 = 3, 2 + 3 = 5, 3 + 5 = 8

<u>Stack / Queue</u> 20 Marks

You are playing a variation of the game Zuma. In this variation of Zuma, there is a single row of colored balls on a board, where each ball can be colored red 'R', yellow 'Y', blue 'B', green 'G', or white 'W'. You also have several colored balls in your hand.

Your goal is to clear all of the balls from the board. On each turn: Pick any ball from your hand and insert it in between two balls in the row or on either end of the row. If there is a group of three or more consecutive balls of the same color, remove the group of balls from the board. If this removal causes more groups of three or more of the same color to form, then continue removing each group until there are none left. If there are no more balls on the board, then you win the game. Repeat this process until you either win or do not have any more balls in your hand.

Given a string board, representing the row of balls on the board, and a string hand, representing the balls in your hand, return the minimum number of balls you have to insert to clear all the balls from the board. If you cannot clear all the balls from the board using the balls in your hand, return -1.

***Example**:*
Input: board = "WRRBBW", hand = "RB"

Output: -1
Explanation: It is impossible to clear all the balls. The best you can do is:
- Insert 'R' so the board becomes WRRRBBW. WRRRBBW -> WBBW.
- Insert 'B' so the board becomes WBBBW. WBBBW -> WW.
There are still balls remaining on the board, and you are out of balls to insert.

Merge/Quick/Radix: 20 Marks

Imagine you're helping a friend put together a giant jigsaw puzzle (the array) with many pieces (elements). However, the pieces are scattered and mixed up! You want to organize them efficiently to make solving the puzzle easier.

Here's your plan:

1. Divide: Split the puzzle pieces into two roughly equal piles (sub-arrays).
2. Conquer: Repeat step 1 recursively on each pile until each pile has only one piece (base case).
3. Combine: Now comes the fun part! Take the two smallest piles (sorted by their "picture") and carefully merge them back together. Compare each piece from both piles one-by-one and place the one that "fits better" (smaller value) into a new, combined, and sorted pile.
4. Repeat: Keep repeating step 3, merging larger and larger sorted piles until all the pieces are combined in the correct order, forming the complete picture (sorted array).

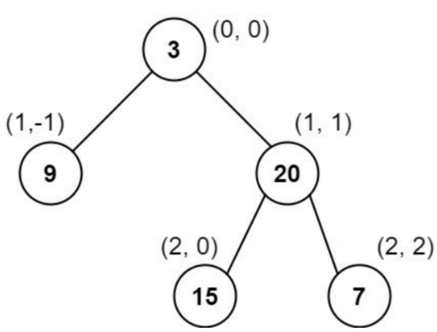
Binary Tree 20 Marks

Given the root of a binary tree, calculate the vertical order traversal of the binary tree.
For each node at position (row, col), its left and right children will be at positions (row + 1, col - 1) and (row + 1, col + 1) respectively. The root of the tree is at (0, 0).

The vertical order traversal of a binary tree is a list of top-to-bottom orderings for each column index starting from the leftmost column and ending on the rightmost column. There may be multiple nodes in the same row and same column. In such a case, sort these nodes by their values.

Return the vertical order traversal of the binary tree.

Example 1:



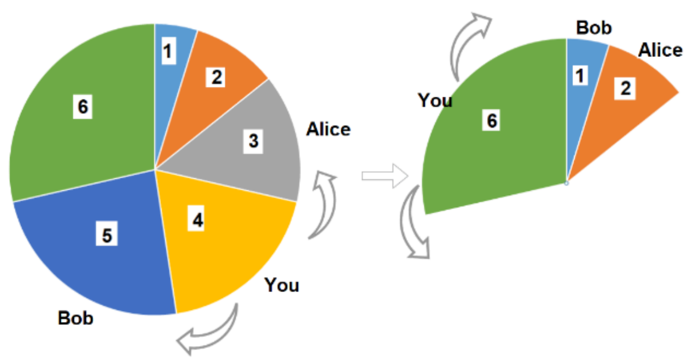
Input: root = [3,9,20,null,null,15,7]
Output: [[9],[3,15],[20],[7]]

Explanation:
Column -1: Only node 9 is in this column.
Column 0: Nodes 3 and 15 are in this column in that order from top to bottom.
Column 1: Only node 20 is in this column.
Column 2: Only node 7 is in this column.

<u>Binary Heap/ Priority Queue:</u> 20 Marks

There is a pizza with 3n slices of varying size, you and your friends will take slices of pizza as follows:

- You will pick any pizza slice.
- Your friend Alice will pick the next slice in the anti-clockwise direction of your pick.
- Your friend Bob will pick the next slice in the clockwise direction of your pick.
- Repeat until there are no more slices of pizzas.
- Given an integer array slices that represent the sizes of the pizza slices in a clockwise direction, return the maximum possible sum of slice sizes that you can pick.



Example:
Input: slices = [1,2,3,4,5,6]
Output: 10
Explanation: Pick pizza slice of size 4, Alice and Bob will pick slices with size 3 and 5 respectively. Then Pick slices with size 6, finally Alice and Bob will pick slice of size 2 and 1 respectively. Total = 4 + 6.