CS 300: ADVANCED PROGRAMMING

SPRING 2024

Module: Haskell

Programming Assignment 1.1

Syed Babar Ali
School of Science and Engineering

Deadline: Feb 4, 2024 (Sunday) 11:55PM

Assignment Guidelines

- The assignment is due at **11:55 PM on Feb 4, 2024 (Sunday)**
- There are a total of **10 questions.**
  - You have to attempt **all of the 5 questions from Easy category**.
  - You have to attempt **4 out of 5 questions from Medium category.**
- Total marks of the assignment are **35**.
- **No imports allowed in any of the questions.**
- We have provided you with a haskell file **"code.hs"** with **function signatures** and **some test cases.**
- Please note that the test cases may not cover all the cases for the questions. There may be other (corner) cases which students have to figure out themselves.
- There are **NO** grace days for any of the assignments in this course.
- There is **NO** late days policy. No submission will be accepted after the deadline.
- **Plagiarism is strictly prohibited.** Students are not allowed to discuss their solutions with others or look over the internet. However, they may seek help from the course staff **ONLY**.
- Students may discuss their queries or clear their confusions about the questions in office hours or on slack (make use of the assignment channel).
- There will be **NO partial** marking for any of the questions. This is being done to be fair to all the students in this course. However, the questions will be graded according to the number of test cases that are passing. For example: if a 10 marks question has 10 test cases and 5 of them are passing for a student, then they will get 5/10 for that question.
- The **submissions will be done using GitHub Classroom**. Your **latest commit** in your assignment repository before the deadline will be considered for the final submission.
- You will have to install hspec which is a testing framework for haskell that we are using for this assignment. It can be simply installed using the following command:
  - "cabal update && cabal install –lib hspec"
- The evaluations may consist of vivas, shortly after the deadline.
- **Start Early & Happy Coding!**

Questions Category: Easy (3 Marks Each)

Question 1: Target Sum Pairs with Ordered Elements

Write a function that takes two inputs: an array and a target value. The function should return a two-dimensional array, where each inner array consists of two numbers, 'a' and 'b', from the original array. These pairs of numbers should satisfy two conditions: their sum equals the target value (a + b = target), and 'a' should be greater than or equal to 'b' (a >= b). The two-dimensional array should be sorted, in ascending order, by the first element.

Sample Input: Array = [4, -1, 0, 1, -3, 2, 5, -6] , target = 1
Output: [[1, 0], [2, -1], [4, -3]]

Question 2: Symmetric Tree

Given a tree, check whether it is a mirror of itself(i.e, symmetric around its root).

Sample Input: Tree = (Node (Node NIL 2 NIL) 1 (Node NIL 2 NIL))
Output: True
Sample Input: Tree = (Node (Node NIL 2 NIL) 1 (Node NIL 3 NIL))
Output: False

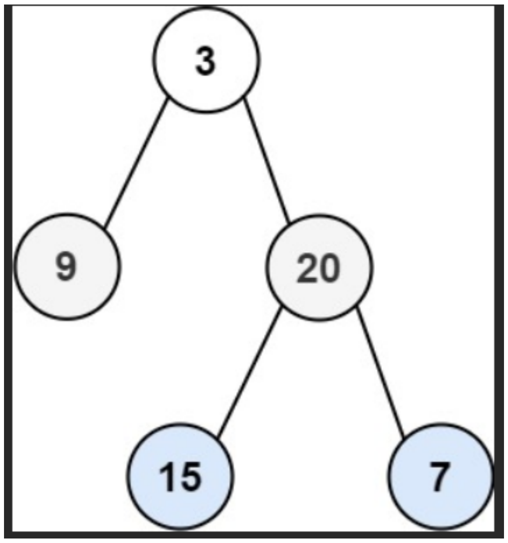
Question 3: Palindrome LinkedList

Given a singly linked list, return true if it is a palindrome or false otherwise.

Sample Input: Linkedlist = (Node 1 (Node 3 (Node 3 (Node 1 Nil))))
Output: = True
Sample Input: (Node 1 (Node 3 Nil))
Output: = False

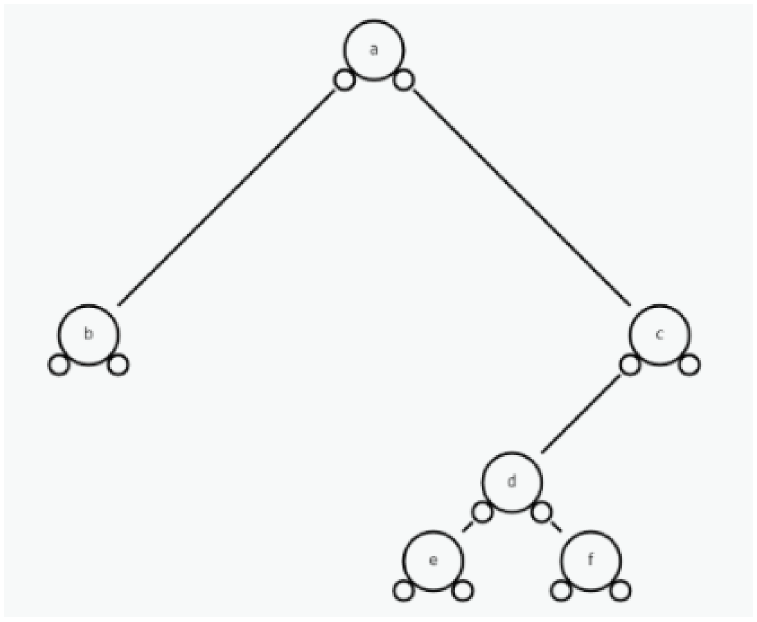
Question 4: Snake Traversal of a Binary Tree

Given a binary tree, return the snake traversal of its nodes' values. (i.e., from left to right, then right to left for the next level and alternate between).



Output: = [3, 20, 9, 15, 7]

Question 5: Binary Tree Construction
You are given a string in the format "ab^cde^f", where each character in the string represents a node in the tree. The "^" character indicates moving up one level in the tree, and any other character represents the value of a node.
Sample Input: "ab^cde^f"
Output:

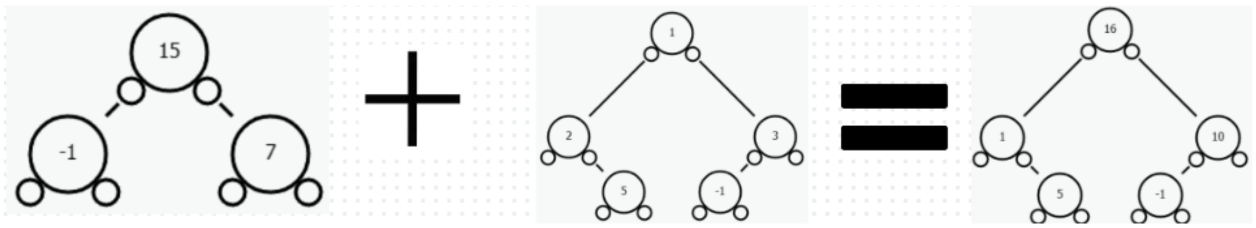


1. Read the first character of the string and create a node with that value. This becomes the root of the tree.
2. For each subsequent character:
   (a) If the character is not “ˆ”, create a new node with that value. Attach this node to the leftmost available position in the current subtree. If the current node already has a left child, move to this left child and consider it as the current node. Continue this process until you find a position where the left child is absent, and then insert the new node there.
   (b) If the character is “ˆ”, move to the right of the current node. The next node you create will be added here, on the right, and this right child node will then become your new current node.
3. Repeat till the end of the string.

Questions Category: Medium (5 Marks Each)

Attempt any 4 of the 5 questions
Question 1:

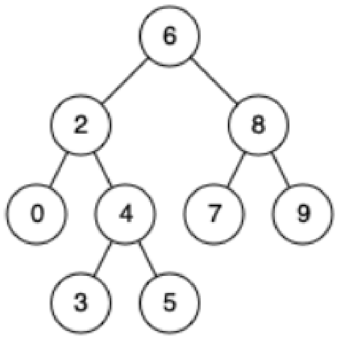
1. **Overloading Addition (+) for Trees**
   Your objective is to overload the ’+’ (addition) operator for tree data structures. This should be done so that the corresponding nodes in two trees are summed up. In cases where a matching node is absent in one of the trees, the existing node should remain as is in the resultant tree. The following example explains how two trees are added.



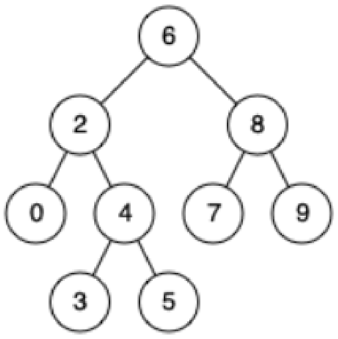
2. **Longest Common Substring**
   In this task, you're provided with two LinkedLists, each holding a string. Your objective is to identify the longest substring that is common to both strings. Once found, this substring should be returned as a LinkedList. If there is no common substring, return Nil.
   Sample Input: str1 = (Node 'a' (Node 'l' (Node 'l' (Node 'o' (Node 'w' Nil)))))
   str2 = (Node 'w' (Node 'o' (Node 'w' Nil)))
   Output: (Node 'o' (Node 'w' Nil))

Question 2: Lowest Common Ancestor of a BST

According to the definition of Lowest Common Ancestor (LCA) on Wikipedia: "The lowest common ancestor is defined between two nodes p and q as the lowest node in T that has both p and q as descendants (where we allow a node to be a descendant of itself)." You are required to find the value of an LCA node of two given nodes in the BST.



P = 4, Q = 7 **Output:** 6

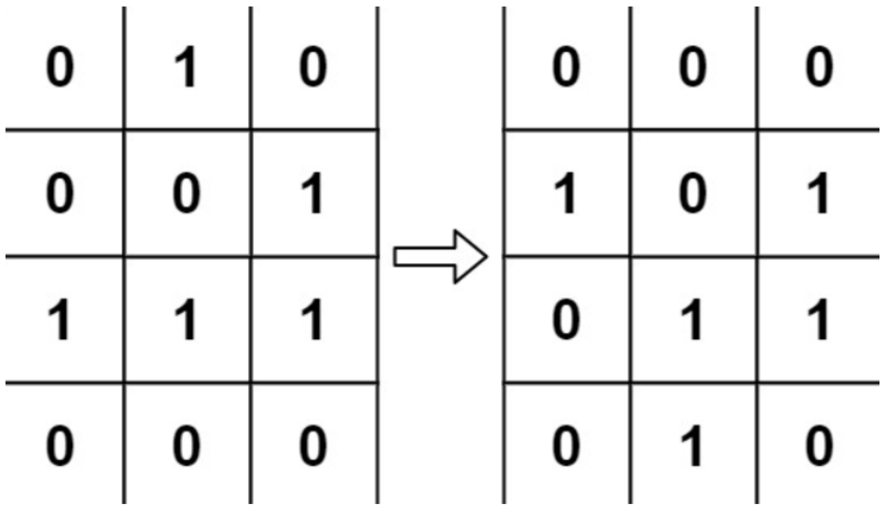


P = 2, Q = 4 **Output:** 2

Question 3: Game of Life
According to Wikipedia's article: "The Game of Life, also known simply as Life, is a cellular automaton devised by the British mathematician John Horton Conway in 1970."
The board is made up of an m x n grid of cells, where each cell has an initial state: live (represented by a 1) or dead (represented by a 0). Each cell interacts with its eight neighbors (horizontal, vertical, diagonal) using the following four rules (taken from the above Wikipedia article):

1. Any live cell with fewer than two live neighbors dies as if caused by under-population.
2. Any live cell with two or three live neighbors lives on to the next generation.
3. Any live cell with more than three live neighbors dies, as if by over-population.
4. Any dead cell with exactly three live neighbors becomes a live cell, as if by reproduction.

The next state is created by applying the above rules simultaneously to every cell in the current state, where births and deaths occur simultaneously. Given the current state of the m x n grid board, return the next state.

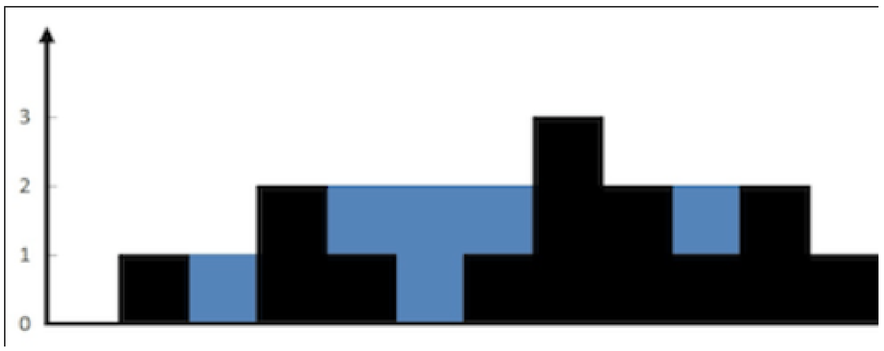


0	1	0
0	0	1
1	1	1
0	0	0

0	0	0
1	0	1
0	1	1
0	1	0

Question 4: Water Collection

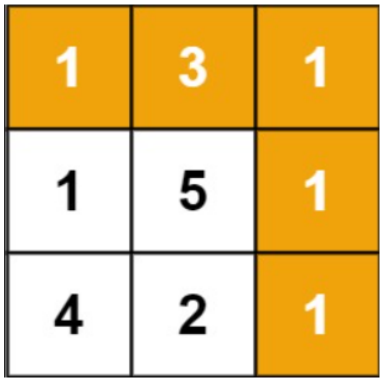
Given n non-negative integers representing an elevation map where the width of each bar is 2, compute how much water it can trap after raining.



Sample Input: Array = [0,1,0,2,1,0,1,3,2,1,2,1]
Output: 12
The above elevation map (black section) is represented by the array [0,1,0,2,1,0,1, 3,2,1,2,1]. In this case, 12 units of rain water (blue section) are being trapped.

Question 5: Minimum Path in a Maze

A maze can be represented as a grid containing paths or multiple paths from the source to the destination. However, for simplicity we will be assuming that we only have NxN grids which will be represented as a 2d List / Array. The grid is filled with non-negative numbers. We will also assume that the start is at list[0][0] and the destination is at list[n-1][n-1]. You are required to find a path from top left (start) to bottom right (destination), which minimizes the sum of all numbers along its path.
Note that you can only move either down or right at any point in time.



1	3	1
1	5	1
4	2	1

Sample Input: Grid = [[1, 3, 1] , [1, 5, 1] , [4, 2, 1]]
Output: 7
