**CS213: Data Structures and Algorithms** **Deadline: 5:00 PM, Mar 19 2021**

# Task #4

*Instructor: Prof. Sharat Chandran*

## 1 Ghastly Checkpoints

Darshan is new to Kargil and wants to explore this locality. Kargil happens to have lots of checkpoints. From each checkpoints there are many ways to go, even (gasp) no way, i.e., you can only return from such special ghastly checkpoints, and can't go any further. Each such checkpoint where you can't go any further dominates Darshan's life and he wants to reach any one such *ghastly* checkpoint (there are many!) as fast as possible. Darshan realizes that checkpoints are placed so that the distance between any adjacent two are equal.

Checkpoints are numbered from 1 to n. You are given how the checkpoints are organised in Kargil. You need to determine the shortest length of the path Darshan needs to take to reach a ghastly checkpoint (the one from which he cannot go further). You can assume that Darshan starts from the hotel checkpoint which is numbered 1.

Hint: Visualise all the routes as a tree with the root of the tree as Darshan's hotel.

**Input format:**
n: No. of checkpoints
The next n-1 lines contains x y where x y represents that there is a road between checkpoints x and y.

**Output format:**
Print the length of shortest path
Output: len
where len is the length of such shortest path!

**Sample Input:**
6
2 4
3 2
2 1
4 6
5 4

**Sample Output:**
2

Reflection Essay:

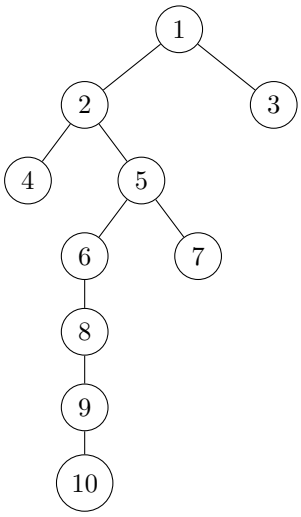
1) Explain how you came up with your solution and its correctness
2) Report the time complexity of you algorithm

## 2 Quad Nodes (Written)

Let T be a binary tree with n nodes. Define a **quad node** to be a node $v$ in T, such that the number of descendants in $v$'s left subtree differ from the number of descendants in $v$'s right subtree by at most 4.

Describe a linear-time method for finding each node $v$ of T, such that $v$ is not a quad node, but all of $v$ descendants (except $v$) are quad nodes.

Here is an example:




Here the quad Nodes are: 10, 9, 8, 7, 6, 5, 4, 3,1

For example, consider node 5: The number of descendants (barring 5) in its left subtree = 4 and number of descendants in its right subtree = 1. Hence it is quad node as abs(4-1) = $3 \leq 4$)

Node(s) which are not quad node but each of its descendants are quad node: 2

Reflection Essay:

Show the working of the given example according to your algorithm. Further

1) Describe your method in full detail and write a pseudo code for it
2) Prove the correctness of your method in brief
3) Derive the time complexity of your method

## 3 Mutant Virus (Written: 20 points)

A virus is identified by a string, that is, a sequence of characters $a_0, a_1, a_2, ..., a_{n-1}$ in an Organic Chemistry notation, i.e., some unknown alphabet (definitely not English). It shows not just the atoms and molecules but the relative 3D structural arrangement such as cis and trans isomers. The size of the alphabet is large but finite and is unrelated to the size of the virus i.e. $n$.

A mutant is a subsequence formed from the original string maintaining the original order. For example, "CNA" is a subsequence (in English) of the string "CORONA" while "CAN" is not.

Empty sequences are not mutants, and the original sequence is also, as expected, not a mutant of itself. Further, though the alphabet is unknown and cannot be indexed, we are allowed a compare function that tells us the ordering of two characters (perhaps using quaternions, but we won't get into that).

Your goal is to compute the total number of distinct mutants. You must use a Binary Search Tree (BST) with the usual `find`, `insert` and `delete` operations.

Explain your algorithm. Expect very little credit if you do not use the BST.

1. High level description of your algorithm. Most efficient algorithm will get full points, other algorithms will get points based on how fast they are.

2. Correctness of your algorithm. Consider using either induction or loop invariants.

3. Implementation Details: Explain clearly what internal data structures one can use (from amongst those discussed so far in the class). Explain how the data structure impacts the time complexity of your solution.

## 4 In summary

Suppose you have to maintain a function (aka map) $f$ from positive integers to positive integers. The map is to be implemented using the concept of Binary Search Trees (**BST**). Thus you need to support the following variants:

- `g x`: returns the value of `f[x]` if `x` exists in map otherwise nothing (null or None).
- `i x y`: Insert `f[x] = y` in the map. If `f[x]` already exists, replace it to `y`.
- `a x1 x2 c`: Add the number `c` to all the elements to the range `x1` to `x2`. This means that all the `f[x]` in the map with `x` in the closed interval `x1` to `x2` must have their values increased by c.
- `d x`: Delete `f[x]` from the map if it exists (i.e., an immediate `g x` returns nothing.)

The goal is these operations must be performed as efficiently as possible. Explain (in your Reflection Essay) the modifications, if any, done to the standard BST described in the lecture and write the complexity of each operation.

**Input:** The input will be a sequence of operations. The map is assumed to be empty at start. Apart from these commands, one extra input `e` is present which denotes the end of the sequence of operations. Your program terminates when it receives `e` as an input.

**Output:** No output when the operations are `i x y`, `a x1 x2 c`, and `d x`. For `g x`, print the value of `f[x]` if it exists. and a new line.

Note: You can implement your node any which way you like as long as there is $O(1)$ information kept in each node.

## 5 Keep Counting

Count the number of binary search trees of $n$ nodes such that its height is between $l$ and $r$, both inclusive. The nodes are numbered from 1 to $n$ and the value associated with node $i$ is $i$ for all $i$ from 1 to $n$. As the answer can be large, report the count modulo 1000000007.

Recall: The height of the root is one.

**Input format:**
$n$: No. of nodes, $l$: Lower bound of the desired tree's height, $r$: Upper bound of the desired tree's height

**Output format:**
The answer

**Sample Input:**
3 1 3

**Sample Output:**
5

Reflection Essay:
1) Draw BST trees for $n = 4, l = 2, r = 4$
2) Explain the working of your algorithm for the above case
3) Explain the overall logic of the algorithm.
4) What is the time and space complexity. Explain.