3120homework02

Class	CPSC 3120
Due Date	@March 5, 2025
Status	In progress
Created time	@February 4, 2025 12:38 AM

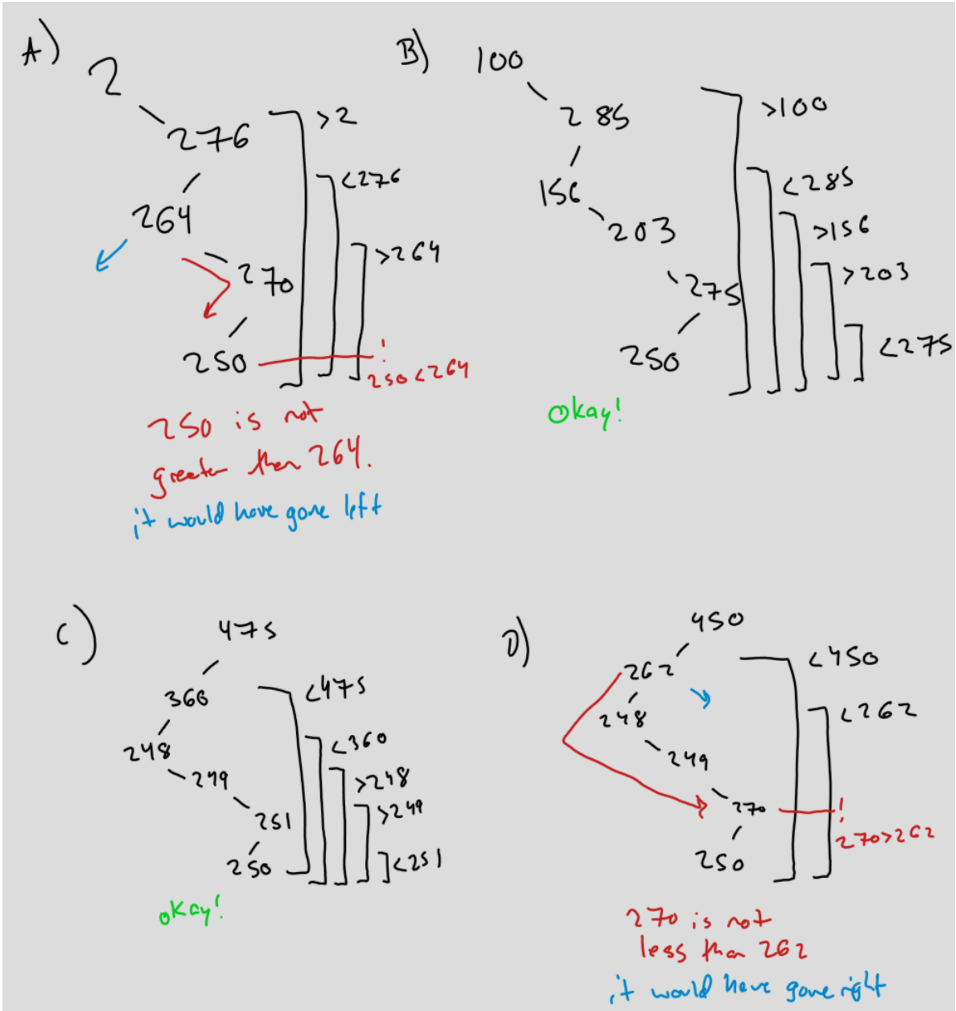
R-3.3 Suppose you have a binary search tree, T, storing numbers in the range from 1 to 500, and you do a search for the integer 250. Which of the following sequences are possible sequences of numbers that were encountered in this search. For the ones that are possible, draw the search path, and, for the ones that are impossible, say why.

a. (2, 276, 264, 270, 250)

b. (100, 285, 156, 203, 275, 250)

c. (475, 360, 248, 249, 251, 250)

d. (450, 262, 248, 249, 270, 250)



R-3.11 Suppose you are given a binary search tree, T, which is constructed by inserting the integers in the set $\{1, 2, \ldots, n\}$ in a random order into T, where all permutations of this set are equally likely. What is the average running time of then performing a select(i) operation on T?

Thanks to this following section from the textbook, they show that the average depth of a node in a randomly constructed BST is O(logn). This is our average running time for select(i).

In addition to the above lemmas, our analysis of a randomly constructed binary search tree also involves harmonic numbers. For any integer $n \geq 1$, we define the nth ***harmonic number***, H_n, as

$$H_n = \sum_{i=1}^{n} \frac{1}{i},$$

for which it is well known that H_n is $O(\log n)$. In fact,

$$\ln n \leq H_n \leq 1 + \ln n,$$

$$E[D(v_j)] \quad \leq \quad H_j + H_{n-j+1} - 1.$$

So, in other words, the expected depth of any node in a randomly constructed binary search tree with n nodes is $O(\log n)$. Or, put another way, the average depth of the nodes in a randomly constructed binary search tree is $O(\log n)$.

Question 3 (10 points)

Prove that for a proper binary tree T with n nodes and heigh h, the total number of nodes is at least $2h + 1$ and at most $2^{h+1} - 1$.

A completely full BST follows the pattern

Level	Nodes
0	1
1	2
2	4
3	8
h	2^h

The total amount of nodes then in a completely full BST would be n = 1+2+4+8+...+2^h or, as a geometric sum, 2^(h+1)-1. This is the upper bound, where the BST has at most 2^(h+1)-1 nodes.

The emptiest a BST can be follows with the root having two children, and for each layer down each child has one child. It follows the pattern

Level	Nodes

0	1
1	3
2	5
3	7
h	2h+1

The tree cannot go lower than 2h+1 nodes, making it the lower bound.

Therefore, the number of nodes is bound by height where

$$2h + 1 \leq n \leq 2^{h+1} - 1$$

Question 4 (10 points)

Prove that for a proper binary tree T with n nodes and height h, the height is at least $\log{(n+1)} - 1$ and at most $\frac{n-1}{2}$.

We can solve for h in terms of n from question 3:

$$2h + 1 \leq n$$
$$2h \leq n - 1$$
$$h \leq \frac{n-1}{2}$$

and

$$n \leq 2^{h+1} - 1$$
$$n + 1 \leq 2^{h+1}$$
$$log_2(n+1) \leq h + 1$$
$$log_2(n+1) - 1 \leq h$$

Therefore, the height is bound by number of nodes where

$$log_2(n+1) - 1 \leq h \leq \frac{n-1}{2}$$