W09 - Binary Search Tree - Treap

Due 10 Mar at 23:59 **Points** 20 **Questions** 8
Available 6 Mar at 8:00 - 10 Mar at 23:59 **Time limit** None
Allowed attempts Unlimited

Instructions

Content and Background

This quiz relates to the content covered in the course up till now. It may also draw upon supporting knowledge and skills expected from a CS sophomore. Please make sure that you are up to date on the coursework before attempting the quiz.

Difficulty

This quiz is equivalent to an in-class exercise. Have pen and paper ready and be prepared to work on challenging problems.

Discussion

Please use Discussion forum to discuss any of the questions. Do not reveal your answers.

Take the quiz again

Attempt history

	Attempt	Time	Score
LATEST	Attempt 1	11 minutes	20 out of 20

Correct answers are hidden.

Score for this attempt: **20** out of 20
Submitted 7 Mar at 22:13
This attempt took 11 minutes.

Question 1

3 / 3 pts

Suppose the ***find*** function is called with some unknown value on a perfect BST that stores values 1,2,...,127.

How many nodes will the ***find*** function have to check in the worst case?

7

How many nodes will the ***find*** function have to check in the best case?

1

Answer 1:

7

Answer 2:

1

Question 2

3 / 3 pts

Suppose a BST is being populated with a 100 values using the ***add*** function. The entire process takes longest when ___________ (check all that apply):

☑ Values are added in ascending order

☑ Values are added in descending order

□ Values are added in ascending order but there are duplicates amongst them

☐ Values are added in perfectly random order

Question 3

2 / 2 pts

The ***remove*** function can be thought of as consisting of a ***find*** part (the *if* and *else if* case), and a part where the meat of the remove happens (the *else* case).

Consider running the ***remove*** function on an arbitrary value. What is the time complexity of the meat of the function (the *else* case) in terms of Big-O?

O(log(n))

Question 4

1 / 1 pts

Consider a Perfect Binary Search Tree with ***n*** nodes. Match the time complexities of the following BST functions:

find	O(log n)
add	O(log n)
remove	O(log n)

Question 5

2 / 2 pts

A new node in a BST is always inserted as a:

○ right child

○ left child

● leaf node

○ internal node

Question 6

5 / 5 pts

```
  1               1               1               1               1               1
   \               \               \               \               \               \
    2               2               3               3               4               4
     \               \             / \             / \             /               /
      3               4           2   4           2   4           2               3
       \             /                                             \             /
        4           3                                               3           2

    2               2               2               2               2               2
   / \             / \             / \             / \             / \             / \
  1   3           1   3           1   3           1   4           1   4           1   4
       \               \               \             /               /               /
        4               4               4           3               3               3

          3               3               3               3               3               3
        /   \           /   \           /   \            /  \            /  \            / \
       1     4         1     4         1     4          2    4          2    4          2   4
        \               \               \              /               /               /
          2               2               2           1               1                1

          4               4               4               4               4               4
         /               /               /               /               /               /
        1               1               2               2               3               3
         \               \             /  \            /  \            /               /
          2               3           1    3          1     3         1               2
           \             /                                             \             /
            3           2                                                2          1
```

Consider the 24 BSTs generated using every permutation of 1,...,4.

1. Consider generating a **randomized BST** by first obtaining a random permutation of 1,...,4 and then inserting those values in that order into the

BST. Such a **randomized BST** will have a height of 2 with probability (round your answer to 3 s.f.) 0.666 .

2. Compute the average (mean) height of the BSTs (round your answer to 3 s.f.): 2.333

Answer 1:

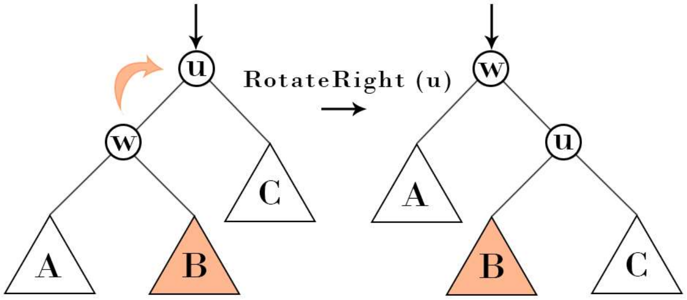
0.666

Answer 2:

2.333

Question 7

2 / 2 pts



The ***rotate_right***(u) / ***rotate_left***(u) helper functions are defined to help in the Treap ***add*** function. What, according to your intuition, should be the time complexity of ***rotate_right*** / ***rotate_left*** ?

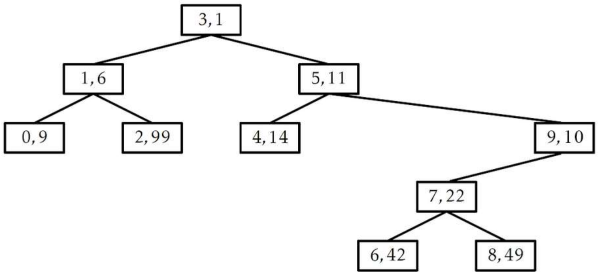
● O(1)

○ O(n)

○ O(log n)

Question 8

2 / 2 pts



The Treap above has one misplaced node**.** What is the **value** of that node? (remember that a node has both a value and a priority)

9

Quiz score: **20** out of 20