Assignment 1

Assignment 1: Lists, Stacks, Queues, Trees

Exercise 1

Implement a Binary Search Tree ADT (called MyIntegerBST) for Integer elements that supports the following operations:

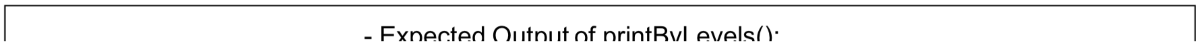
– public void insert(Integer value). //Insertion operation seen during the lectures

– public Integer mostSimilarValue(Integer value)

- If "value" has been inserted in the tree, returns "value"
- Otherwise, returns the value "s0" such that
    1. s0 has been previously inserted in the tree
    2. for any other value "s1" in the tree, | s1-value | >= | s0-value |
    - Note: | x-y | means the absolute value of operation x-y

– public void printByLevels()

- Prints all elements at depth "d" in the tree before elements at depth "d+1". It Starts with the root (depth=0) as "Depth 0: rootValue". It continues printing the children of the root (depth=1), as "Depth 1: ...". It continues with the root grandchildren (depth=2), etc.
- Next figure shows an example of a tree and the expected output.



Your class must implement the interface provided in file "A1Tree.java".

Implement all operations as fast as you can, and attach a pdf document where you analyze the worst-case time complexity of operations mostSimilarValue and printByLevels. The grade of the exercise will consider the quality of the implementation and the correctness of the analysis.

You can assume that only positive integers will be inserted.

Exercise 2

Propose the implementation of an ADT "SequenceWithMinimum" that supports the following operations:

– public void insertRight(Integer value)

- It inserts the value on the right of the sequence.
- Example:

    - SequenceWithMinimum seq= new SequenceWithMinimum() //seq={}
    - seq.insertRight(5) //seq={5}
    - seq.insertRight(4) //seq={5,4}
    - seq.insertRight(15) //seq={5,4,15}

– public Integer removeRight()

- It removes the rightmost value in the sequence
- Example (continues from previous seq={5,4,15}):
    - seq.removeRight() //returns 15, and seq={5,4}

– public void insertLeft(Integer value)

- It inserts the value on the left of the sequence.
- Example (continues from previous seq={5,4}):
    - seq.insertLeft(9) //seq={9,5,4}

– public Integer removeLeft()

- It removes the leftmost value in the sequence
- Example (continues from previous seq={9,5,4}):
    - seq.removeLeft() //returns 9, and seq={5,4}
    - seq.removeLeft() //returns 5, and seq={4}

– public Integer findMinimum()

- It returns the minimum value in the sequence
- Example (being seq={5,4,15})
    - seq.findMinimum() // returns 4, and seq={5,4,15} (seq remains unchanged)

Submit a pdf document where you propose the data structure and the implementation of operations that execute all the 5 operations as fast as you can. Add an analyis of the time complexity of your operations. The grade of the exercise will consider the quality of the implementation and the correctness of the analysis. Minimum to get a "pass" in this exercise: worst-case of the 4 insert/remove operations O(1) and worst-case of findMiniumum O(N). Hint: you can reach amortized O(1) for all the 5 operations.

Exercise 3

Implement your solution proposed in Exercise 2. Your class should implement the interface provided in file "A1SequenceWithMinimum.java". The grade of the exercise will consider the measured execution time (in my computer) in an average case and in the worst-case.

Important: You must implement the ADTs from scratch without reusing data structures from libraries (none among java lists, stacks, queues, trees, etc.). This applies to all exercises in this assignment.

- assignment1AADS.zip 26 September 2018, 2:01 AM

Submission status

This assignment will accept submissions from **Tuesday, 24 September 2019, 12:00 AM**

Attempt number	This is attempt 1.
Submission status	No attempt
Grading status	Not graded
Due date	Thursday, 3 October 2019, 11:55 PM

Time remaining	10 days 2 hours
Last modified	-
Submission comments	Comments (0)

◄ E3: Priority Queues & Sorting

Previous exams ►