CSCI 230 – Homework 4

Stacks, Queues, Trees

Part 1: Stacks & Queues

Objectives

- Implement a deque with O(1) complexity.
- Implement methods for BinarySearchTree.
- Test the implementation thoroughly with tests that cover all possible use cases.

Background information

Queue, Stack, Binary Trees from Shaffer textbook

Assignment

Part 1: Stacks and Queues

A **deque** is a data structure consisting of a list of items, on which the following operations are possible:

push(x): Insert item x on the front end of the deque.
pop(): Remove the front item from the deque and return it.
inject(x): Insert item x on the rear end of the deque.
eject(): Remove the rear item from the deque and return it.

Write routines to support the deque that take O(1) time per operation.

Part 2: Trees

Add the following methods to the BinarySearchTree BST class provided by the OpenDSA textbook:

1. Implement a binary search tree member method that does a level-order traversal. This method prints a tree one level per line. Name your method **printTreeLevelOrder**. It should only generate output, requires no input parameter or return type. Feel free to use the Java Queue interface and LinkedList or other class for the queue needed to accomplish this task. Within BinarySearchTree class one way that you can implement a Queue using the Java Collection Framework:
   Queue<Node> queue = new LinkedList<Node>; Note: read the documentation for the

Queue interface(https://docs.oracle.com/javase/8/docs/api/java/util/Queue.html), the methods used are add (instead of traditional enqueue), remove (instead of the traditional dequeue), peek etc. Note, since Queue extends Collection, it inherits isEmpty.

2. Implement a recursive member method called **countParentsOfOne**, that returns the number (int) of nodes that have exactly one child. The user should call a public method with same name that takes no parameters; the public method calls the private recursive method that takes the root as parameter.

What to submit:

- A zipped folder containing all of your java files **and no subdirectories/folders**.
- One of the files must contain the **deque** class in a file naturally named **deque.java** The filename and class name must match exactly what is in bold.
- The second file must contain the BST class from OpenDSA with the additional methods. Name the file **BST.java**.
- One class named **HW4** stored in a file named **HW4.java** that contains a main method that demos your ordered list and what works and specifies in comments what doesn't.
- Any other java files required of your solution. // There might not be any.
- Put deque.java, BST.java, HW4.java, and any other java files needed in a directory named YourLastNameFirstInitial_HW4. Then zip that directory. Then you will have a folder YourLastNameFirstInitial_HW4.zip. This is what you need to submit for HW4.