CSDS 233 Due: October 23, 2022, 11:59 PM

Assignment 3

General instruction: This assignment includes two parts, written and programming. Please write/type your answers neatly so they can be readable. For the written part, please show the steps of your work. Please submit a single PDF file for the written part and a zip file for the programming part before 11:59 PM - October 23, 2022.

Special office hour for this assignment will be on Friday October 14, 2022, 6:00pm-7:00pm in Zoom (Meeting ID: 536 484 7715 Passcode: 048217), on which day I will extend my office hour to 8:00pm. Feel free to ask question via email: yxf484@case.edu.

Written Problems (50 pts)

1. AVL Trees (25 pts)

a. Insert 26, 200, 39, 17, 43, 5, 51, 22, 19 to an empty AVL tree in the given order. Show the final tree after all insertions. **(5 points**)

b. Add one or two elements such that it causes a single right rotation in the tree. State the added number and show the final tree after the insertion. Moreover, do the same job causing a single left rotation. **(5 + 5 = 10 points)**

c. Insert 27 to the AVL tree from the (b)'s result after ***right rotation***. Properly rebalance the resulting tree. Now, delete one element such that it causes a single left rotation, right rotation, or double rotation in the tree. State the deleted number and the rotation type and show the final tree after the deletion operation. **(5 points)**

Answer the following questions, which are independent from the part (a) to the part (c):

d. What is the minimum number of nodes to build an AVL tree of height 5? **(2 points)**

e. Every AVL tree is balanced. (True or False) **(1 points)**

f. Every AVL tree is complete. (True or False) **(1 points)**

g. Some AVL trees are complete. (True or False) **(1 points)**

2. B Trees (15 pts)

a. Construct a B-Tree by using 4, 3, 9, 2, 7, 11, 1, 6, 8, 11, having minimum degree t = 2. Show the final tree after all insertions. **(5 points**)

b. Delete element 11 from your B-Tree. Show the final tree after the deletion. **(5 points)**

c. What are the time complexities of B-Tree insertion and deletion operations? Explain how you achieved the answer. **(5 points)**

3. Heap (10 pts)

Assume we build a binary max-on-top heap from keys 10, 22, 100, 34, 62, 5, 8, 45, 3, 72, 7, 44, 11, 13, 2.

a. Show the result of building this heap by inserting the above keys one at a time in the order given (from left to right), into an initially empty binary heap. Show the final tree after all insertions. **(5 points)**

b. Show the heap from part (a) above after executing three removeMax operations on this heap. Please show key steps. **(5 points)**

Programming Problems (50 pts)

Binary Search Tree (50 points)

Create a class named BST_Class with the following methods:

a) void insert(int key) – inserts a node in the binary search tree (BST)
b) void postorder(Node node) – post-order traversal of BST
c) void inorder(Node node) – in-order traversal of BST
d) void preOrder(Node node) – pre-order traversal of BST
e) boolean search(int key) – checks whether a given key exists in BST
f) int minValue(Node root) – finds the smallest element in BST
g) Void deleteKey(int key) – deletes a node from BST

You need to develop this class as efficiently as possible. You are allowed to use the provided sample code, or you could also start from scratch. The sample code will realize (a) insert and (c) in-order methods, which, I believe, could make you have a good start. Now you should finish the remaining tasks.

In your *Main* class, create an instance of *BST_Class* and realize the following outputs.

```
The BST Created with input data(Left-root-right):
7 10 12 45 50 90
The BST after Delete 12(leaf node):
7 10 45 50 90
The BST after Delete 90 (node with 1 child):
7 10 45 50
The BST after Delete 45 (Node with two children):
7 10 50
Key 50 found in BST: true
Key 12 found in BST: false
BST => PreOrder Traversal:
45 10 7 12 90
BST => InOrder Traversal:
7 10 12 45 90
BST => PostOrder Traversal:
7 12 10 90 45
```

Submission

The submissions will be evaluated on completeness, correctness, and clarity. Please provide sufficient comments in your source code to help the TAs read it. Please generate a single zip file containing all your *.java files needed for this assignment and optionally a README.txt file with an explanation about added classes and extra changes you may have done. Name your file ‘P3_YourCaseID_YourLastName.zip’. Submit your zip file electronically to Canvas. Notice: Before your programming assignment is graded, TAs will **feed other examples** to your code as input, so please make sure your program works for any input before your submission. You can test your code by feeding multiple inputs and check them if they are correct.

Assignment 3

Grading:

- **Implementation: 30 pts**
- **Test: 10 pts**
- **Design and style (Proper encapsulation/information hiding) 10 pts**