CIS 3501 Data Structures

Program 2

Winter 2019

Due Dates: February 19 (Part 1)

February 26, 2019 (Parts 1 & 2, source code, input/output files)

For this assignment, implement search trees from scratch (i.e. no use of predefined coding routines to perform tree actions).

You will be creating and comparing operations on BST, AVL and Splay trees.

There will be a separate output file for each type of tree.

Reading from a file (file doesn’t exist?, empty? – print message and continue), insert a series of numbers into each of the trees, printing the tree to a file and to the screen each time, and the tree height after each insert. A set of numbers are listed below, and you need to insert those numbers into your trees, in that order. The beginning of your output should look similar to the output below.

NOTE: this is only ONE test scenario ….. for full credit you need to create others to cover all the test scenarios

The printTree() function can display your tree in whatever way you choose, the tree needs to be easy to read. Show the tree, and tree height, after every insert.

The trees should show one after the other on the screen.

Example output after first 6 inserts for BST ...

Insert 9

9

Tree height is: 0

Insert 7

9

7

Tree height is: 1

Insert 14

9

7 14

Tree height is: 1

Insert 25

9

7 14

25

Tree height is: 2

Insert 2

9

7 14

2 25

Tree height is: 2

Insert 23

9

7 14

2 25

23

Tree height is: 3

The AVL Tree will need to show, after every insert, that it rebalances correctly and maintains the structure of an AVL tree. You will need to implement functions for the rotations: LL, RR, LR, RL.

The Splay Tree will need to show after every insert that the new node rotates to the root. Implement functions for the rotations: Zig, Zag, Zig-Zig, Zag-Zag, Zig-Zag, Zag-Zig.

For the output of the AVL and Splay Trees you also need to list the rotations performed to create the new tree.

NOTE: expect to see each rotation combination shown in test cases – i.e. what does tree look like, what value is inserted, what does tree look like after insert and rotation(s) used.

Your program will now process a file of searches, inserts and deletes from the trees.

The file will contain an action code: S – search, I – insert, and D- delete (what if it isn’t one of those?????) and the number for that action

NOTE: Account for twin chain ……. just add duplicate node to twin chain

Display the action and value and what tree would look like after the action takes place. You are to print the number of operations to perform the action.

Keep track of the actions and the number of operations in a 2-dimensional array (row = action, columns = BST, AVL, Splay). When no more actions print the table and sum the columns by search, insert, delete and overall. Write a report summarizing your results.

NOTE: Remember a Splay Tree should work best if 80% of the accesses are to 20% of the items. Should you have different tests to prove this?????

Use turn-in template for document submission (only need samples of your output in screenshots to show functionality is working as the entire run should be in the output files.

Required Test Scenario:

7

14

25

2

23

4

18

10

27

3

22

30

24

26

21

1

28

8

19

16

17

12

29

6

5

15

20

13

11

S 11

S 26

S 1

S 7

I 17

D 22

S 11

D 16

I 15

S 29

S 17

S 11

D 17

D 25

D 16

I 33

S 17

S 16