Data Structure Lab

Assignment-6

Date of Assignment: 6- Sept -2017 **Date of Submission: 14- Sept -2017**

Treaps are introduced in 1989 by Aragon and Seidel. A treap *T* is a binary tree with each node storing two values: a *key* (take it to be a positive integer) and a *priority* (a floating-point value in the range [0, 1). In addition, there are three pointers in each node: left, right and parent, with the usual meanings. The tree *T* is a binary search tree with respect to the key values. Moreover, the priority values must obey the max-heap ordering property. *T* is not assumed to be full, that is, the heap structure property is not enforced. We only require each node to store a priority value greater than or equal to the priority values of its two child nodes.

Insert into a treap: Let *T* be a treap, and we want to insert a key *x* with a priority *y* in *T*. Initially, we follow the standard BST insertion procedure to insert *x* in *T*. If *x* is already present in *T*, no change is made in *T* (even when the new priority *y* of *x* is different from its old priority). Now, we adjust the priority values along the unique path from the inserted leaf to the root node. Let *p* be a node on this path, and *q* be its parent. If *q* is NULL, or the priority of *q* is greater than or equal to the priority of *p*, we are done. Otherwise, if *p* is the left child of *q*, we make a right rotation at *q*. Finally, if *p* is the right child of *q*, we make a left rotation at *q*. This single rotation restores both BST and heap orderings at *q*. However, heap ordering may be violated at the parent of *q*. So we continue our adjustment procedure further up in the tree.

Delete from a treap: We start by locating the key *x* to be deleted. If *T* does not contain *x*, no change is made. So assume that *x* is present at a node *p*. Three cases may occur:

Case 1: If node is a leaf, delete it.

Case 2:If node has one child NULL and other as non-NULL, replace node with the non-empty child.

Case 3: If node has both children as non-NULL, find max of left and right children.

Case 3.a: If priority of right child is greater, perform left rotation at node.

Case 3.b: If priority of left child is greater, perform right rotation at node.

The idea of case 3 is to move the node to down so that we end up with either case 1 or case 2.

Write a *main*() function that does the following tasks:

1. Start with an initially empty treap *T*.

2. Read the number *n* of keys to be inserted in *T*.

3. Read *n* (key, priority) pairs. These are inserted one by one in *T*. Print *T* after each insertion.

4. Read the number *m* of deletions.

5. Read *m* keys. These key values are deleted one by one from *T*, and *T* is printed after each deletion.

Sample Output

The following transcript shows one insertion followed by one deletion. The (key, priority) pairs are printed.

```
(58, 0.935971) -> (38, 0.731085), (90, 0.651462)
(38, 0.731085) -> (16, 0.435779), (50, 0.500000)
(16, 0.435779) -> (NULL,-), (28, 0.138100)
(28, 0.138100) -> (NULL,-), (NULL,-)
(50, 0.500000) -> (NULL,-), (53, 0.282950)
(53, 0.282950) -> (NULL,-), (NULL,-)
(90, 0.651462) -> (86, 0.287194), (NULL,-)
(86, 0.287194) -> (73, 0.201614), (NULL,-)
(73, 0.201614) -> (NULL,-), (NULL,-)
Number of nodes = 9
+++ insert(63,0.993582)
(63, 0.993582) -> (58, 0.935971), (90, 0.651462)
(58, 0.935971) -> (38, 0.731085), (NULL,-)
(38, 0.731085) -> (16, 0.435779), (50, 0.500000)
(16, 0.435779) -> (NULL,-), (28, 0.138100)
(28, 0.138100) -> (NULL,-), (NULL,-)
(50, 0.500000) -> (NULL,-), (53, 0.282950)
(53, 0.282950) -> (NULL,-), (NULL,-)
(90, 0.651462) -> (86, 0.287194), (NULL,-)
(86, 0.287194) -> (73, 0.201614), (NULL,-)
(73, 0.201614) -> (NULL,-), (NULL,-)
Number of nodes = 10
+++ delete(63)
(58, 0.935971) -> (38, 0.731085), (90, 0.651462)
(38, 0.731085) -> (16, 0.435779), (50, 0.500000)
(16, 0.435779) -> (NULL,-), (28, 0.138100)
(28, 0.138100) -> (NULL,-), (NULL,-)
(50, 0.500000) -> (NULL,-), (53, 0.282950)
```

```
(53, 0.282950) -> (NULL,-), (NULL,-)

(90, 0.651462) -> (86, 0.287194), (NULL,-)

(86, 0.287194) -> (73, 0.201614), (NULL,-)

(73, 0.201614) -> (NULL,-), (NULL,-)

Number of nodes = 9
```

Submission Guideline

```
If (your roll number is between 16CS01001 and 16CS01022)
        Email to ARVIND (vp14)
else
        Email to RUPESH (se10)
```