Lab 09 – Min Heap of Integers

# Problem

Assigned the task of creating a heap (minimum) of integers. Needed to code methods to insert a new node into the heap, delete a node from the heap, heapsort (make a copy of the heap, remove and print each element from the clone) and also a print method for the heap.

# Proposed Solution

Create a minimum heap tree of integers, with the minimum value always being on the top. Insert a new element to the very bottom of the heap (in breadth order) and then bubble the node up to its necessary location by swapping nodes. The delete heap removes the very first element (root) from the heap, and then re-sorts the heap into a balanced min heap again. HeapSort just removes each element in order (from a copy) and prints out the sorted heap. Print method merely prints the elements in index (breadth) order.

# Tests and Results

Was able to code all methods to work properly. Insert adds new elements to the heap, print heap prints the nodes in breadth order, heapsort clones the heap and prints them numbers out in sorted order, and the delete method removes the root and resorts the tree.

# Problems Encountered

Didn’t have many issues with this lab, I had some trouble with the delete element due to an error in my code: my for each loop (for (int i : heap) ) wasn’t working for some reason, and when I recoded the for loop as a general for loop (**for** (**int** i = 0; i < size; i++ ) ) this fixed my issue.

# Conclusions and Discussions

I think this was a good example of a heap. I feel that inserting nodes and deleting nodes in this data structure is easy, and the bubble up / down methods are relatively easy to understand.

# Additional Questions

**Lab Report Questions:**

1. . Demonstrate each step of inserting the elements 21, 27, 49, 11, 23, 1, 13, 16, 33, and 17 into this min heap.

Step One: Root

21

Step Two: Root

21

27

Step Three: Root

21

27 49

Step Four: Root

21

27 49

11 (Bubble UP)

Root

21

11 49

27

Root

11

21 49

27

Step Five: Root

11

21 49

27 23

Step Six: Root

11

21 49

27 23 1 (Bubble UP)

: Root

11

21 1

27 23 49

: Root

1

21 11

27 23 49

Step Seven: Root

1

21 11

27 23 49 13

Step Eight: Root

1

21 11

27 23 49 13

16 (Bubble UP)

Root

1

21 11

16 23 49 13

27

Root

1

16 11

21 23 49 13

27

Step Nine: Root

1

16 11

21 23 49 13

27 33

Step Ten: Root

1

16 11

21 23 49 13

27 33 17 Bubble Up

Root

1

16 11

21 17 49 13

1. 33 23

2. Demonstrate each step of inserting the elements 21, 27, 49, 11, 23, 1, 13, 16, 33, and 17 into this max heap.

(SAME BUBBLE UP TECHNIQUES AS ABOVE, ONLY NOW THE ROOT IS A MAX)

Step One: Root

21

Step Two: Root

27

21

Step Three: Root

49

21 27

Step Four: Root

49

21 27

11

Step Five: Root

49

23 27

11 21

Step Six: Root

49

23 27

11 21 1

Step Seven: Root

49

23 27

11 21 1 13

Step Eight: Root

49

23 27

16 21 1 13

11

Step Nine: : Root

49

33 27

23 21 1 13

11 16

Step Ten: Root

49

33 27

23 21 1 13

11 16 17

1. Show each step when deleting two elements from the min heap.

Root

1

16 11

21 17 49 13

1. 33 23

-Delete Root-

Root

23

16 11

21 17 49 13

27 33 (Bubble Down)

Root

16

23 11

21 17 49 13

27 33

Root

16

21 11

23 17 49 13

27 33

Root

16

17 11

23 21 49 13

27 33

Root

11

17 16

23 21 49 13

27 33

Root

11

17 13

23 21 49 16

27 33

-Delete Root-

Root

33

17 13

23 21 49 16

27 Bubble Down

Root

17

33 13

23 21 49 16

27

Root

17

23 13

33 21 49 16

27

Root

17

23 13

27 21 49 16

33

Root

17

21 13

27 23 49 16

33

Root

13

21 17

27 23 49 16

33

Root

13

21 16

27 23 49 17

33