HW2 - ADS

Saturday, November 22, 2025   7:01 PM

CUS 1151 – Advanced Data Structures Assignment
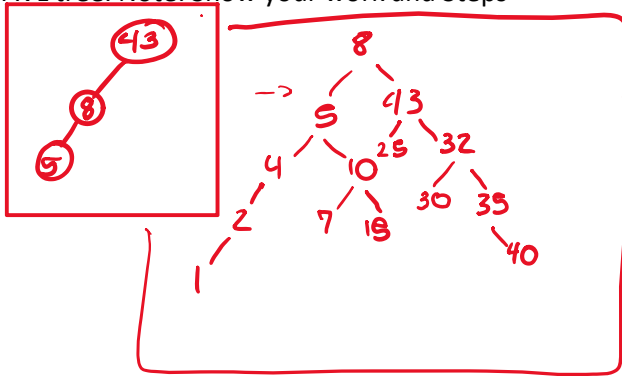Student Name: Darajah Harriott
Assignment: HW2
Instructor: Dr. Fazel Keshtkar
HW2 is based on AVL Trees and Heap Trees

The project is divided into two main parts, which involves analyzing operations of data structures like balanced AVL Trees and priority queues (Min/Max Heap Trees), and Part C, which requires building a Java application to process a large text file. The Java application will use built-in tools to clean, tokenize, count word frequencies, and analyze the data.
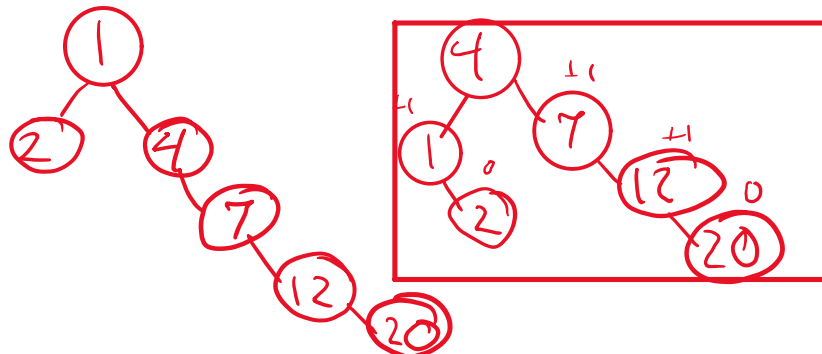
Part A) BST and AVL trees [20 points]:

Q1) Show the result of inserting 43, 8, 5, 10, 4, 7, 15, 25, 32, 2, 1, 30, 35, and 40 in that order into an initially empty AVL tree. Note: Show your work and Steps

Q2) Suppose we want an AVL tree with a height of 5 and a minimum number of nodes;

1) draw such a tree and explain your steps (Nodes can be with empty as long as each node is balanced)

2) how many minimum nodes do we node to have an AVL tree with height 5?
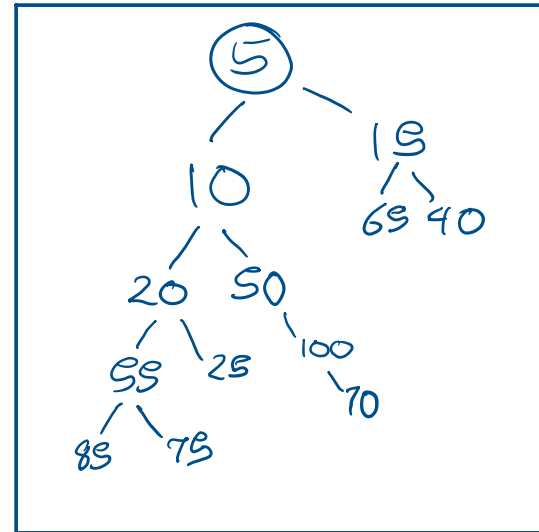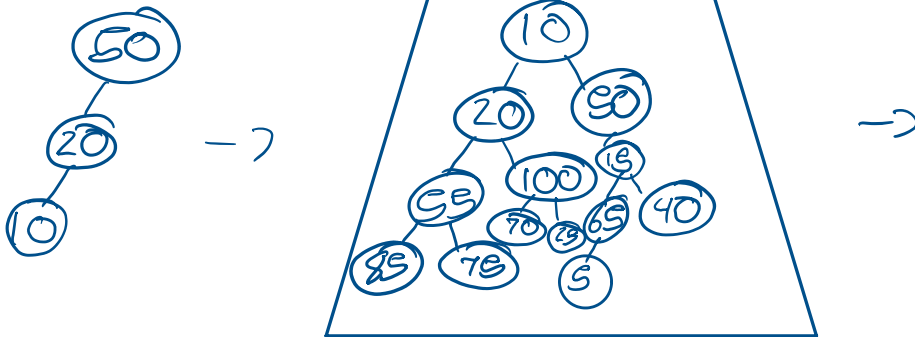The minimum number of nodes required for an AVL tree of height 5 is 20.

Part B) Heap trees [20 points]: Note: Show your work for all questions
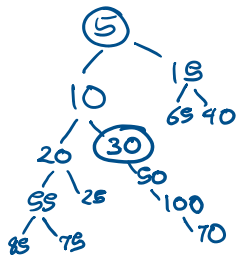
Q3)

1) Draw the Min-Heap tree from the following keys (entered from left to right):
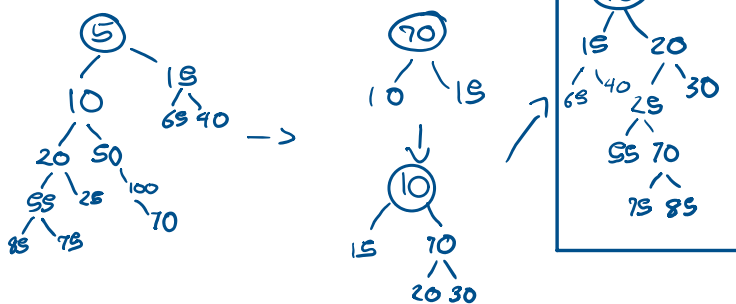
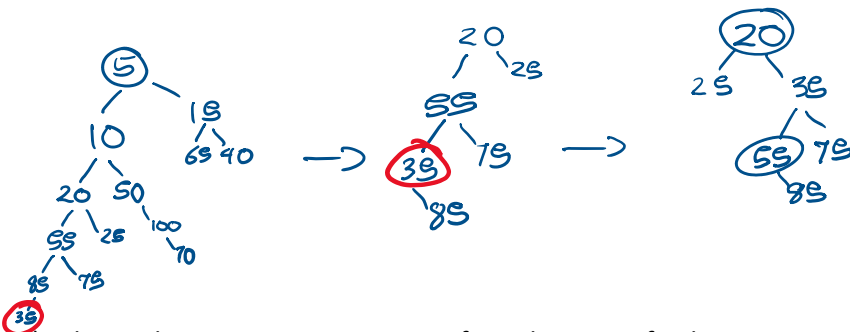50, 20, 10, 55, 100, 15, 65, 40, 85, 75, 70, 25, 5

2) Insert 30 in a final tree in part 1.

3) Delete-min from a final tree in part 2

4) Insert 35 to a final tree in part 3

5) What is the array representation of min-heap in a final tree in **part 4**
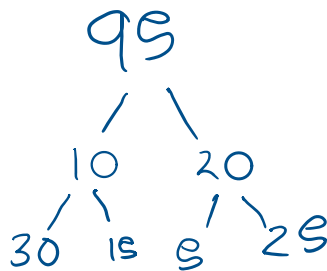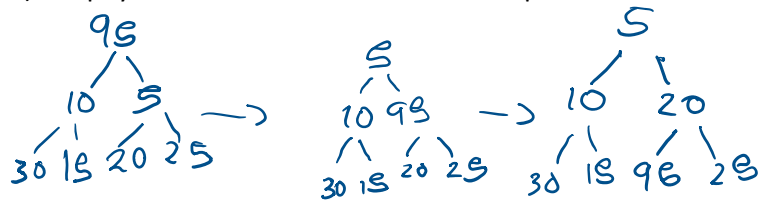
[10, 15, 20, 65, 40, 50, 25, 55, 75, 85, 70, 100, 30]

Q4)

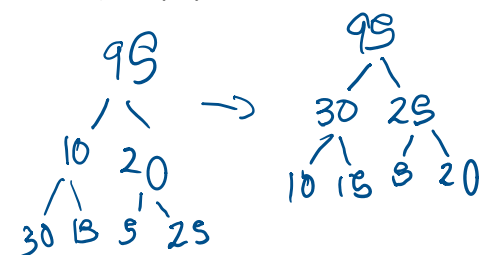Given the binary tree below, and answer the questions:

Given the binary tree below, and answer the questions:

```
          95
         /  \
       10    20
      / \   /  \
    30  15 8    25
```

1) Heapify the above tree to make Min-heap

```
       95                      8                       5
      /  \                    / \                     / \
    10    5     -->         10  95      -->         10    20
   / \   / \               / \  / \                / \   /  \
 30 15 20 25             30 15 20 25             30 15 95  25
```

2) Heapify the above tree to make Max-heap

```
       95                      95
      /  \          -->       /  \
    10    20                30    25
   / \   / \               / \   /  \
 30 13 5  25             10 15  8   20
```

3) Show the array representation of the final tree in Part 1.

[5, 10, 20, 30, 15, 95, 25]

```
            (5)
           /   \
         10     20
        / \    /  \
      30  15  95   25
```

4) Show the array representation of the final tree in Part 2.
[95, 30, 25, 10, 15, 5, 20]

```
            (95)
           /    \
         30      25
        / \     /  \
      10  15   20   5
```