CS-202

Fundamental Structures of Computer Science II

Section: 1

Homework 3

Mehmet Hasat Serinkan

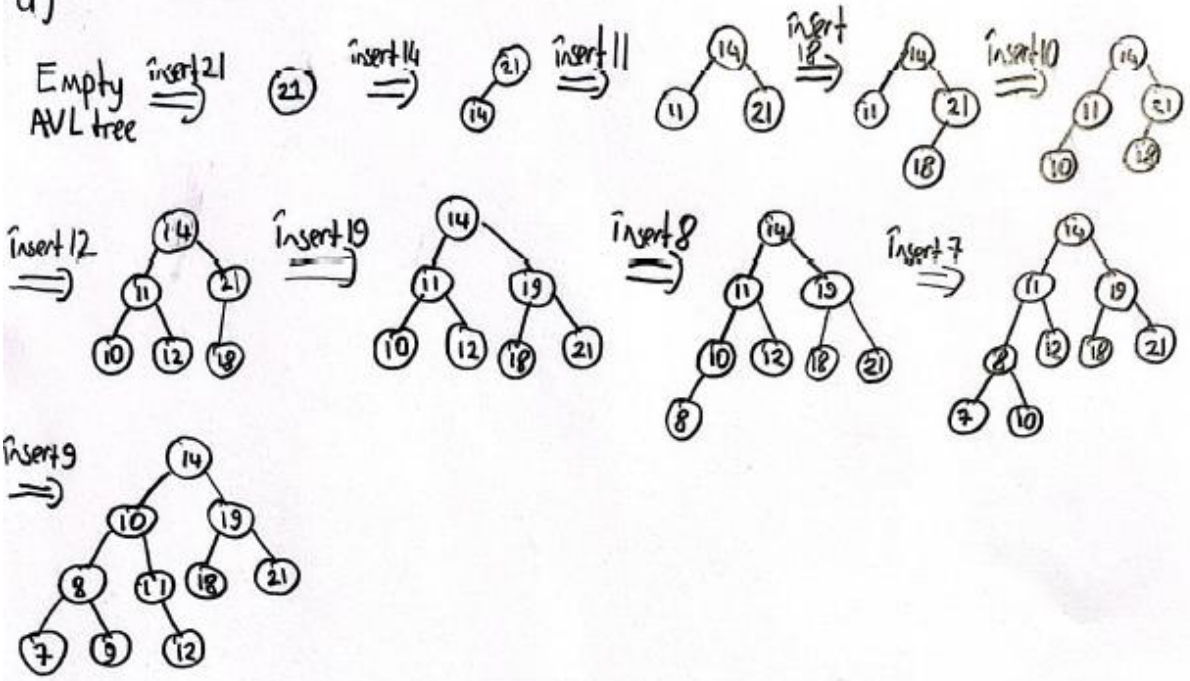
21901649

21.07.2022

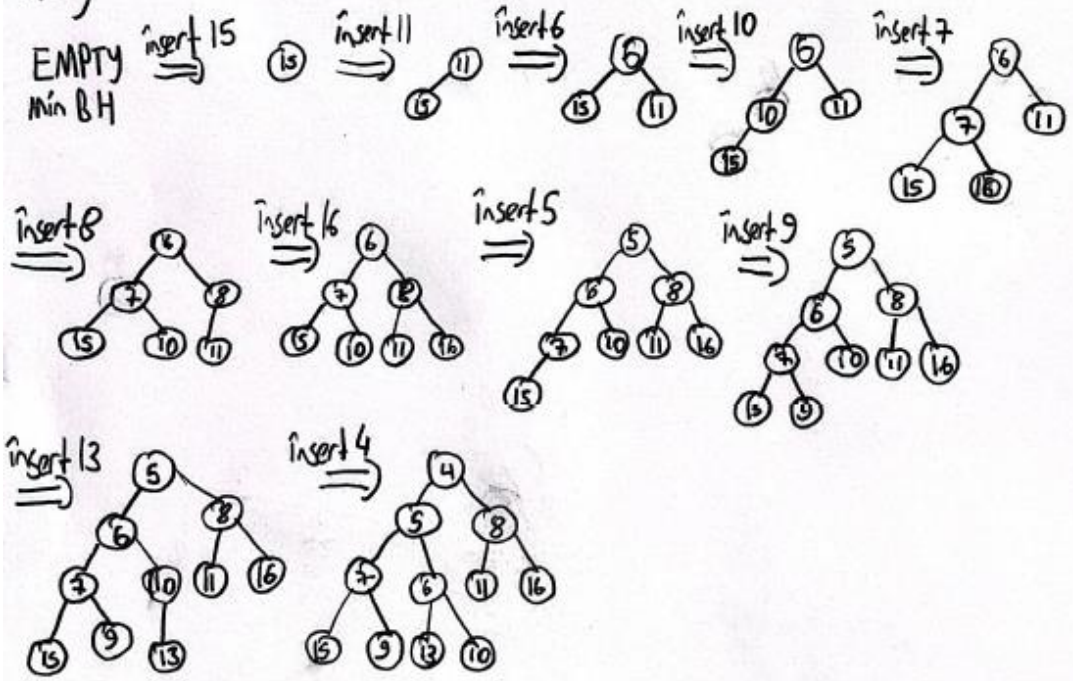
Question 1)

Question 1-

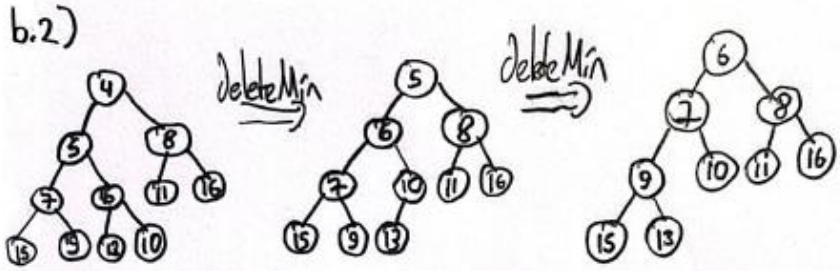
a)



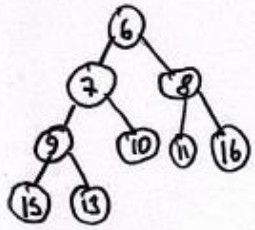
b.1)



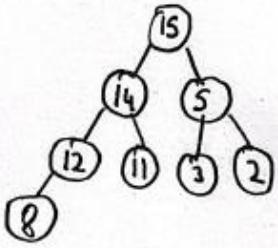
b.2)



c) MIN HEAP

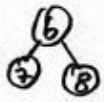


MAX HEAP



Pre-Order: 6, 7, 9, 15, 13, 10, 8, 11, 16
In-Order: 15, 9, 13, 7, 10, 6, 11, 8, 16
Post-Order: 15, 13, 9, 10, 7, 11, 16, 8, 6

Pre-Order: 15, 14, 12, 8, 11, 5, 3, 2
In-Order: 8, 12, 14, 11, 15, 3, 5, 2
Post-Order: 8, 12, 11, 14, 3, 2, 5, 15

For a minheap, pre-order traversal can give a sorted list. Ex: 
However, post-order traversal cannot give a sorted list.

For a max-heap, post-order traversal can give a sorted list. Ex: 
However, pre-order traversal cannot give a sorted list.

The in-order traversal cannot give a sorted list because the root will be have highest or lowest value than remaining ones.

d.1)

The minimum number of nodes in an AVL tree of height h is;

$$\text{MinNodes} = \begin{cases} 0 & \text{if } h=0 \\ 1 & \text{if } h=1 \\ 2 & \text{if } h=2 \\ \text{MinNodes}(h-1) + \text{MinNodes}(h-2) + 1 & \text{if } h>2 \end{cases}$$

d.2) 17710

e)

```
bool minHeapChecker(Node* root, int index, int n)
    if(root = NULL)
        return true
    if(index >= n)
        return false
    if ((root->left != NULL AND root->item >= root->left->item) OR
        (root->right != NULL AND root->item >= root->right->item))
        return false
    return minHeapChecker(root->left, 2*index+1, n) AND minHeapChecker(root->right,
        2*i+2, n)
```

Question 2)

Heap data structure is a complete binary tree whether:

- It is empty or
- Its root contains an item greater than its children or
- Its root contains an item smaller than its children.

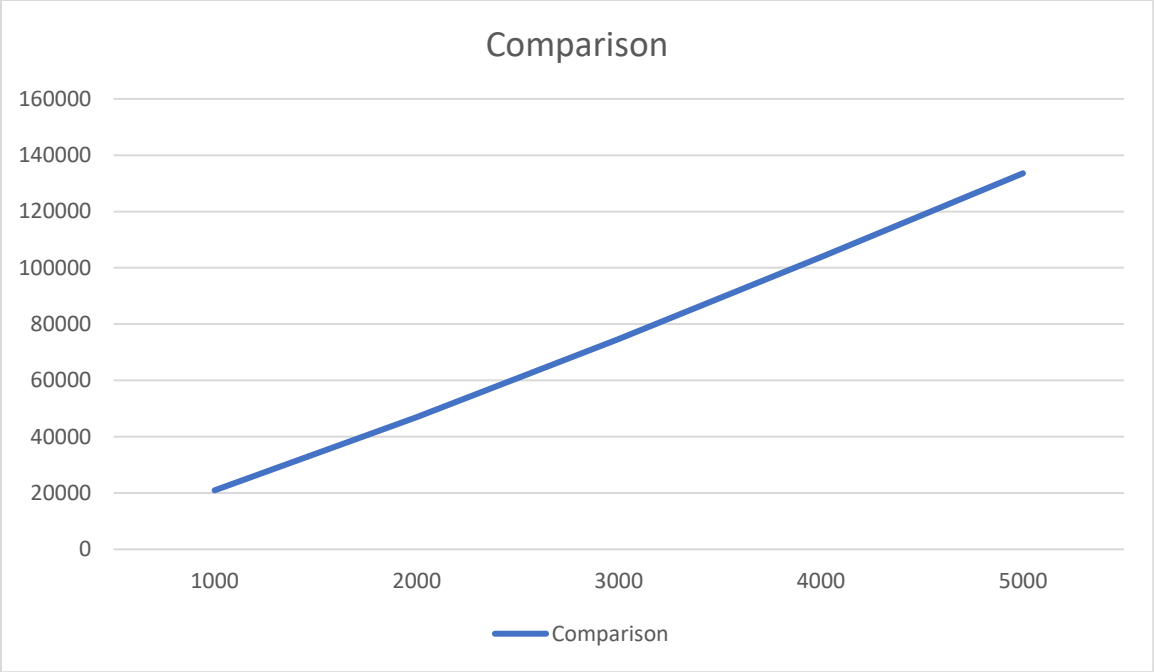
If the root's item is greater than its children, then it is a maxheap.

If the root's item is smaller than its children, then it is a minheap.

Binary heaps are completely filled on all levels except possibly the lowest level. The lowest level is filled from left to right.

File	Size	Key Comparison
data1	1000	20948
data2	2000	46981
data3	3000	74650
data4	4000	103755
data5	5000	133565

Size – Comparison Graph



The time complexity for heapRebuild is O(logN). In the heapsort function, it calls heapRebuild for N/2 – 1 times. Then, the sorting part of heapsort takes O(nLogN) times. The total time complexity is O(nLogN).

It can be seen from the graph that the key complexity for heapsort function is O(nLogN).