Rojin and Suresh Assignment 6

1. Given a binary tree T containing n keys and a key k, design a recursive pseudo-code algorithm FindSmallerKeys(T, k) that returns a Sequence of keys in T that are less than or equal to key k.

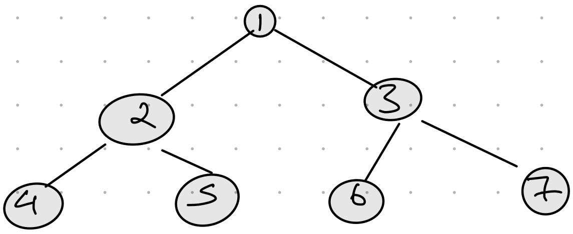
```
// Find Min Keys with Tail Recursion

Input: Binary tree Tree , key: min key
Output: Array with less than or equal key

Algorithm FindSmallerKeys(Tree, key) // Tree: Tree, key: number
    smallerKeys: number[] = [];      //array
    keyHelper(Tree, key, smallerKeys, Tree.root())
    return smallerKeys;

Algorithm keyHelper(Tree, key, smallerKeys, node)
    if Tree.isExternal(node) then
        return;

    if node.element <= key then
        smallerKeys.push(node.element);
        return;

    keyHelper(Tree, key, smallerKeys, Tree.leftChild(node))
    keyHelper(Tree, key, smallerKeys, Tree.rightChild(node))
```

2. Suppose a binary tree T is implemented using an array S, as described in the notes. If n items are stored in S in sorted order, starting with index 1, is the tree T a heap? Justify your answer.

Since, all children are greater then there parents, Array S sorted in ascending order can be used as Min-Heap. As shown in example below,

Sorted array in ascending order: [null, 1, 2, 3, 4, 5, 6, 7].

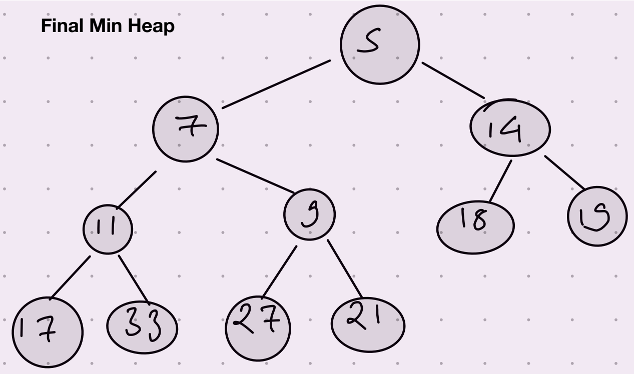
array s = [ø, 1 , 2, 3, 4, 5 , 6, 7]



Construction of Min Heap

3. Building a Min-heap

Keys: 9 11 14 5 21 18 19 17 33 27 7

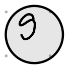

Final Min Heap

array

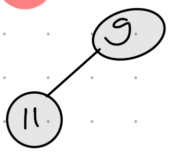
∅	5	7	14	11	9	18	19	17	33	27	21
0	1	2	3	4	5	6	7	8	9	10	11

2i	2i+1
2x1	2x1+1
2	3
Right	left

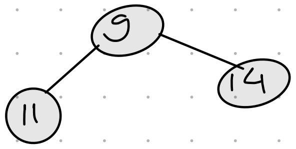
1 Insert 9, Since tree is empty. 9 becomes root.



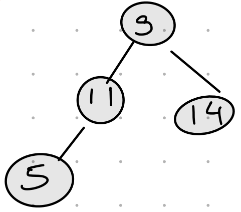
2 Insert 11 as Left Child of 9.



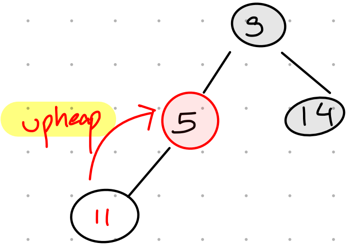
3 Insert 14 as Right Child of 9.



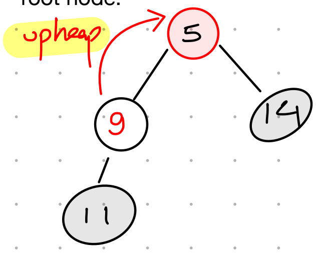
4 Insert 5 as Left Child of 11. It voilet the Min Heap Property, we have to restore by doing Upheap.



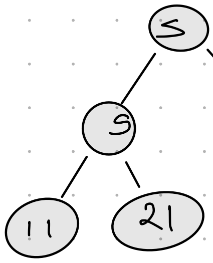
5 Performing upheap but still violets heap property. Next upheap 5 with root 9.



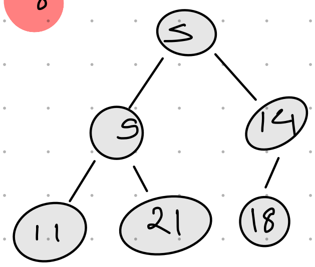
6 Performing upheap to make 5 our new root node.



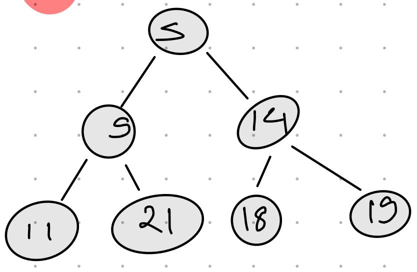
7 Inserting 21, as 9 right child.



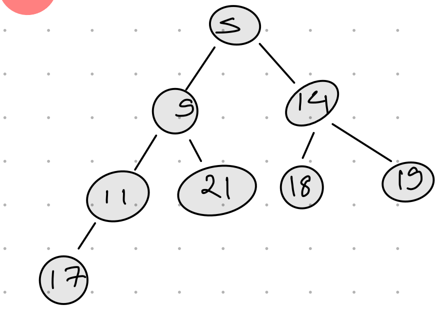
8 Insert 18 as Left Child of 14.



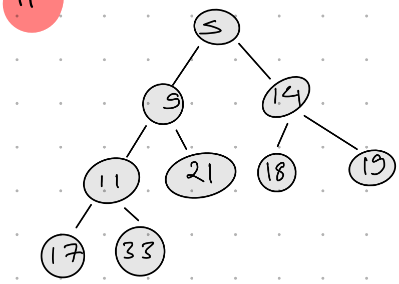
9 Insert 18 as Right Child of 14.



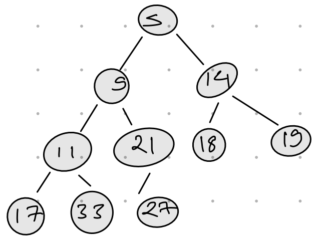
10 Insert 17 as Left Child of 11.



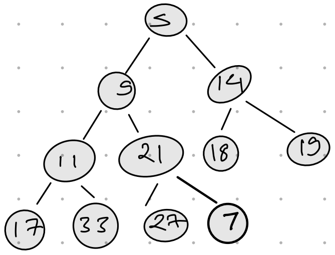
11 Insert 33 as Right Child of 11.



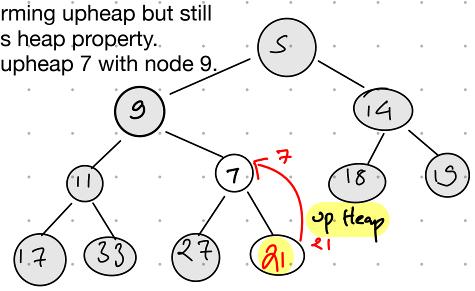
12 Insert 27 as Left Child of 21.



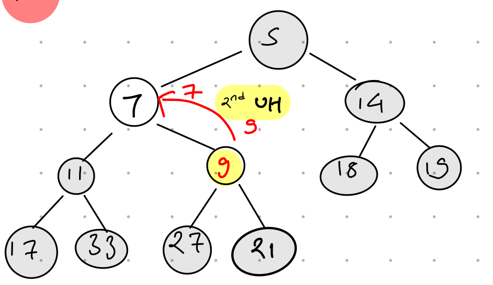
13 Insert 7 as Right Child of 21. It voilet's the Min Heap Property, we have to restore by performing Upheap.



14 Performing upheap but still violets heap property. Next upheap 7 with node 9.



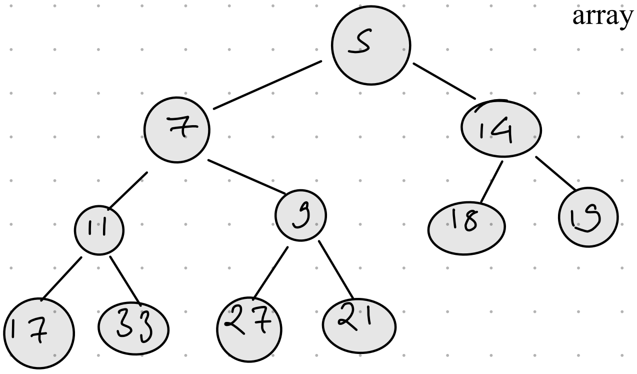
15 Performing Upheap 7 with node 9.



b) Remove the min key from the heap, apply the procedure you learned in the class. Finally show the array representation of the heap.

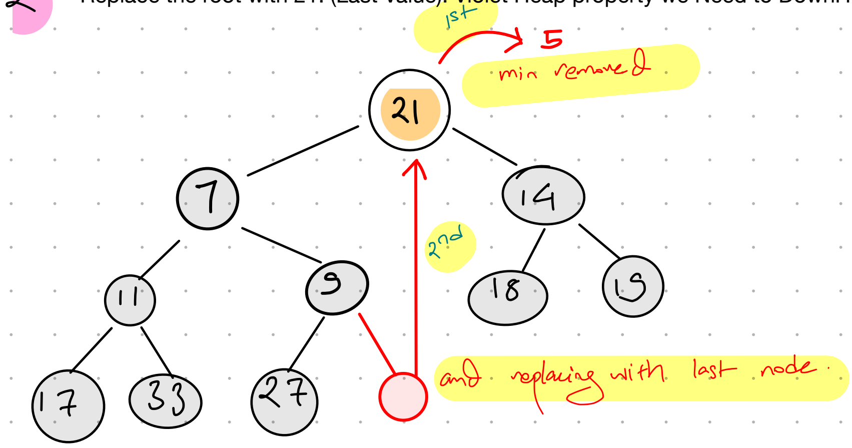
1

RemoveMin( ) —> 5



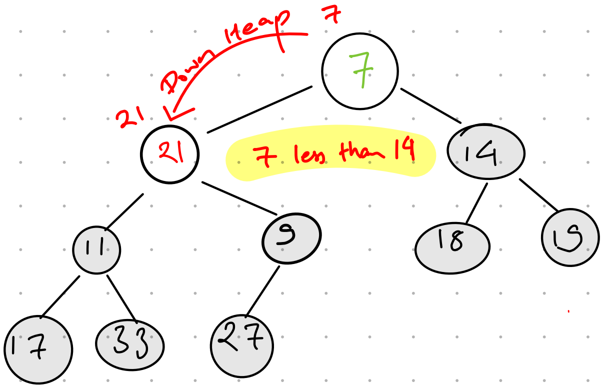
2

Replace the root with 21. (Last Value). Violet Heap property we Need to DownHeap.



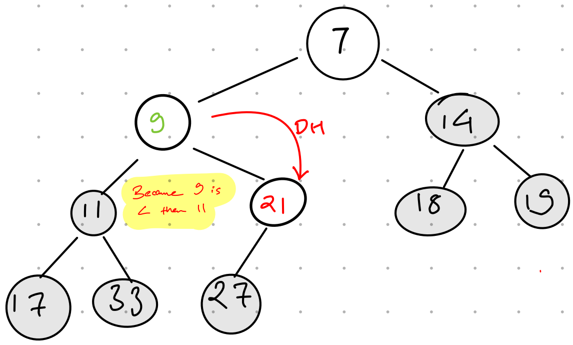
3

Performing DownHeap to left child because 7 is less then 14.
But, Still violets Heap property. We need to downheap again.

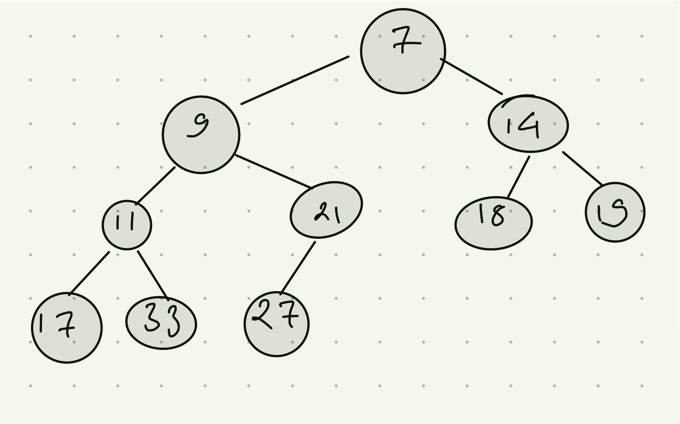


4

Performing DownHeap 21 with 9. Because 9 is less then 11.



Final Min Heap


7
9
14
11
21
18
19
17
33
27