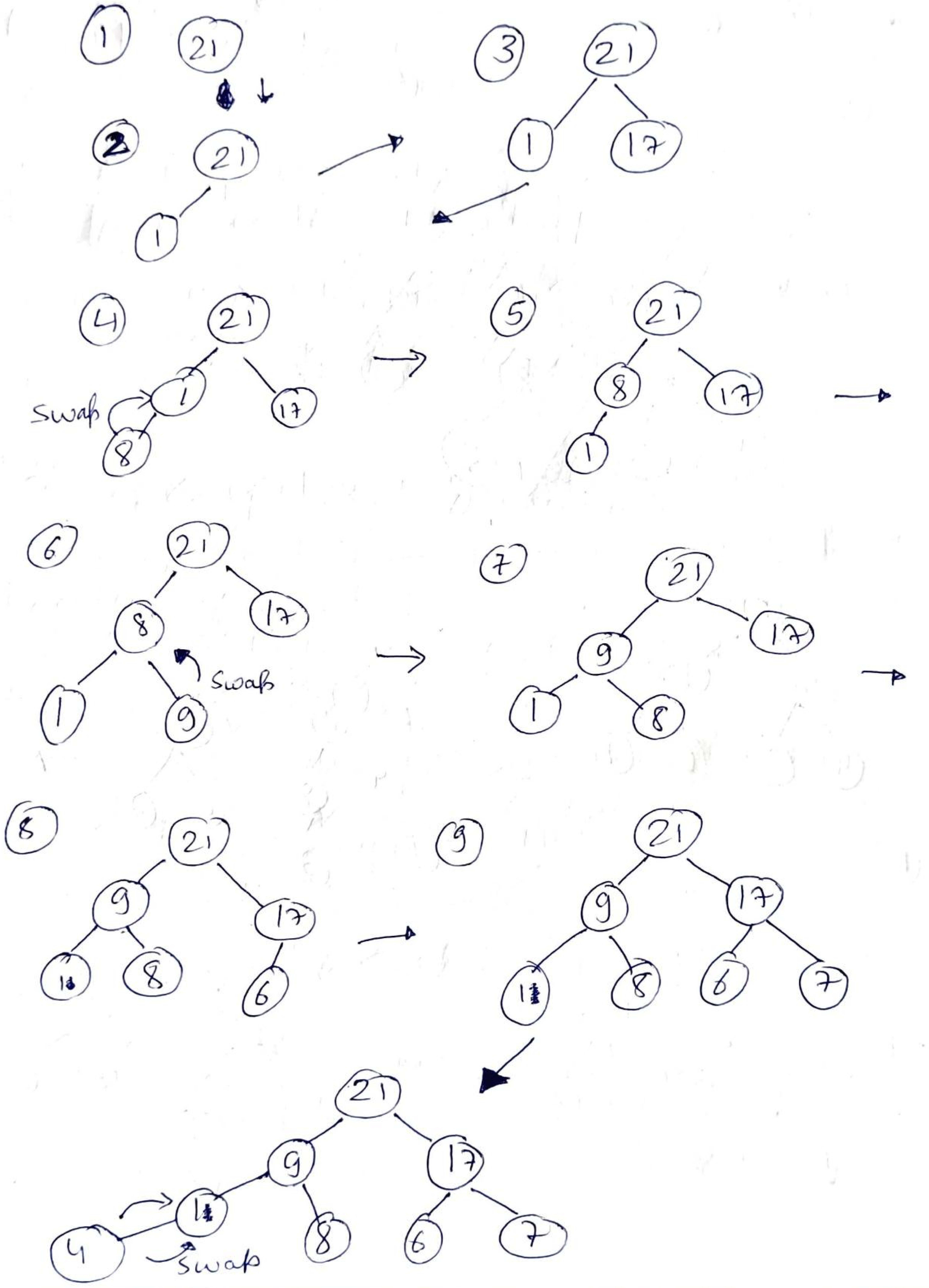
Ankit Raj | DAA - Test | 1906534

Ans 1) 21 , 1, 17, 8, 9, 6, 7, 4, 3, 8, 5



Scanned by TapScanner

Ankit Raj DAA - Test 1906534

① Max-Hep (A)

```
{
    A - heap- size = A - lngth
    for ( i = floor (A. length (2) to1)
        max_ heapify (A, i );
}
```

Time Complexity = O(n log n'

Produre

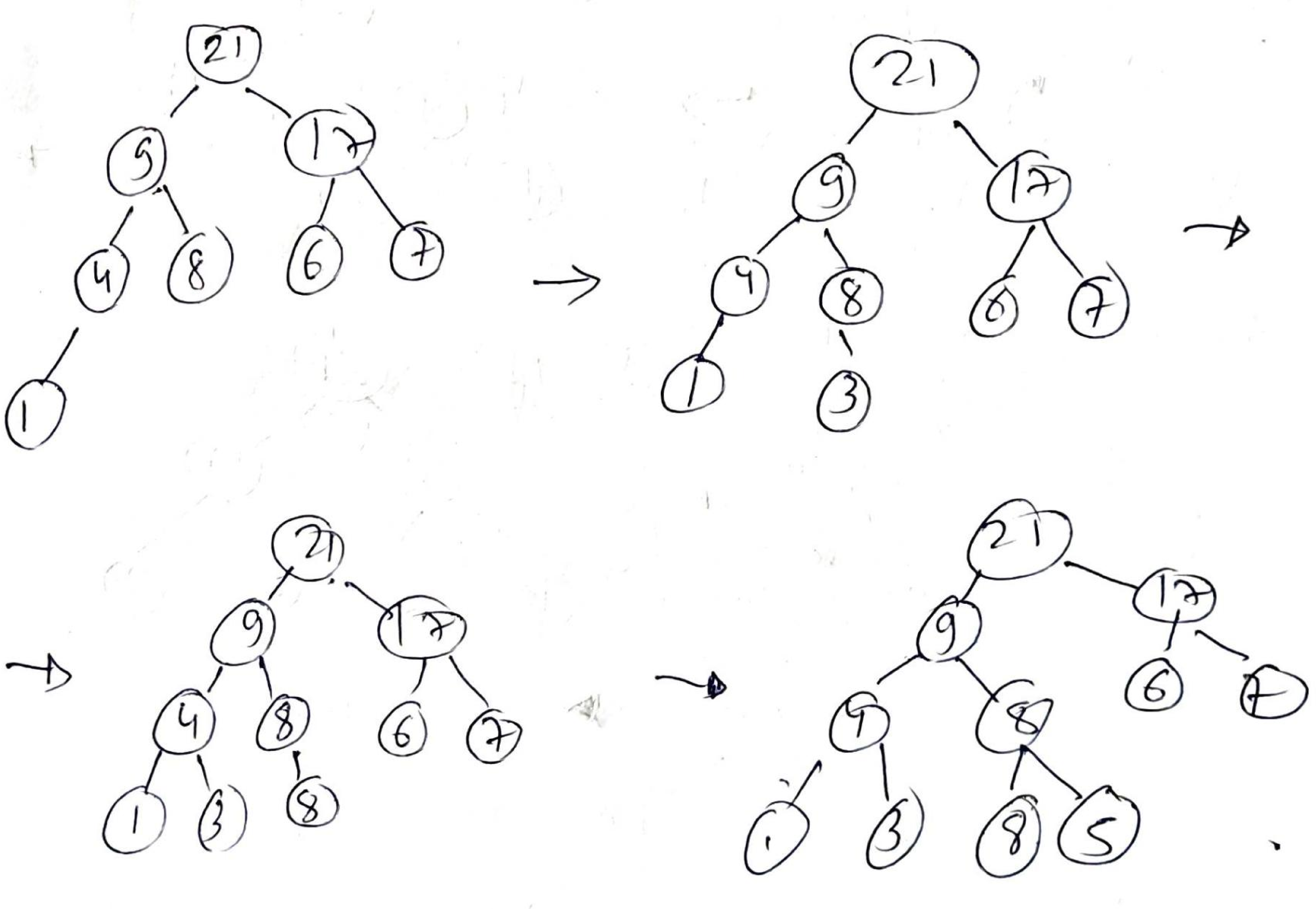
① Create a new node in the heap.

② Assign value to node

③ compare values of child node with parent node
If parent < child then swap

④ If parent > child, then normally we increase string with first left and then right.

⑤ Repet Steps ③ ⑧ ④ ~~till~~ until heap property holds.


21
9
17
4
8
6
7
1
21
9
17
4
8
6
7
1
3
21
9
17
4
8
6
7
1
3
8
21
17
9
6
7
4
8
1
3
8
5

② ⓐ PARTITION (A, p, r)

```
x = A [r]
i = p - 1
for j = p to r - 1
    if A[j] <= x
        i = i + 1
        exchange A[i] with A[j]
exchange A[i+1] with A[r]
return i+1
```

when all are eaual then loop terminate
k = r - 1, So value of q = k + 1 = r.

- PARTITION'(A, P, r) //modified

```
x = A [r]
k = p - 1
l = p - 1
for j = p to r - 1
    if A[j] < x
        k = k + 1
        exchange A[j] with A[k]
        l = l + 1
```

~~Else~~

```
else if A[j] = x
        l = l+1
        exchange A[j] with A[l]
exchange A[l+1] with A[r]
return (A, [(l+k)/2]+1)
```

(3) The basic idea of the greedy abbroch is to calculati the ratio value weight of each item and sort the item on basis of this ratio. Then take the item with height ratio & add them until. the condition reaches of weight or other condition.