R-2. **Answer:**

|  |
| --- |
| Step 1 : **22**, 15, 26, 44, 10, **3**, 9, 13, 29, 25  Step 2 : 3, **15**, 26, 44, 10, 22, **9**, 13, 29, 25  Step 3: 3, 9, **26**, 44, **10**, 22, 15, 13, 29, 25  Step 4: 3, 9, 10, **44**, 26, 22, 15, **13**, 29, 25  Step 5: 3, 9, 10, 13, **26**, 22, **15**, 44, 29, 25  Step 6: 3, 9, 10, 13, 15, **22**, 26, 44, 29, 25  Step 7: 3, 9, 10, 13, 15, 22, 26, 44, 29, 25  Step 8: 3, 9, 10, 13, 15, 22, 26, 44, 29, 25  Step 9: 3, 9, 10, 13, 15, 22, 25, 44, 29, 26  Step 10: 3, 9, 10, 13, 15, 22, 25,26 29, 44  Step 11: 3, 9, 10, 13, 15, 22, 25,26 29, 44  Step 12: 3, 9, 10, 13, 15, 22, 25,26 29, 44 |

Selection sort find minimum element and swap it with the first element of unsorted section.

Runtime of finding minimum element= O(n).So, for n element the run time is = O(n2).

R-2.9 **Answer:**

|  |
| --- |
| Step 1 : 22, 15, 26, 44, 10, **3**, 9, 13, 29, 25  Step 2 : **15**, 22, 26, 44, 10, **3**, 9, 13, 29, 25  Step 3 : **10**,15, 22, 26, 44, **3**, 9, 13, 29, 25  Step 4 : **3**, 10,15, 22, 26, 44, 9, 13, 29, 25  Step 5 : 3, **9**, 10,15, 22, 26, 44, 13, 29, 25  Step 6 : 3, 9, 10, **13**, 15, 22, 26, 44, 29, 25  Step 7 : 3, 9, 10, 13, 15, 22, 26, **29**, 44, 25  Step 8 : 3, 9, 10, 13, 15, 22, **25**, 26, 29, 44  Step 9 : 3, 9, 10, 13, 15, 22, 25, 26, 29, 44 |

In insertion sort, we take an unsorted element and swap it to left until it find a proper position, which takes O(n) time. So, for n element, it takes O(n2) time.

R-2.10 **Answer:**

44, 29, 26, 25, 22, 15, 13, 10, 9, 3

R-2.13 **Answer:**

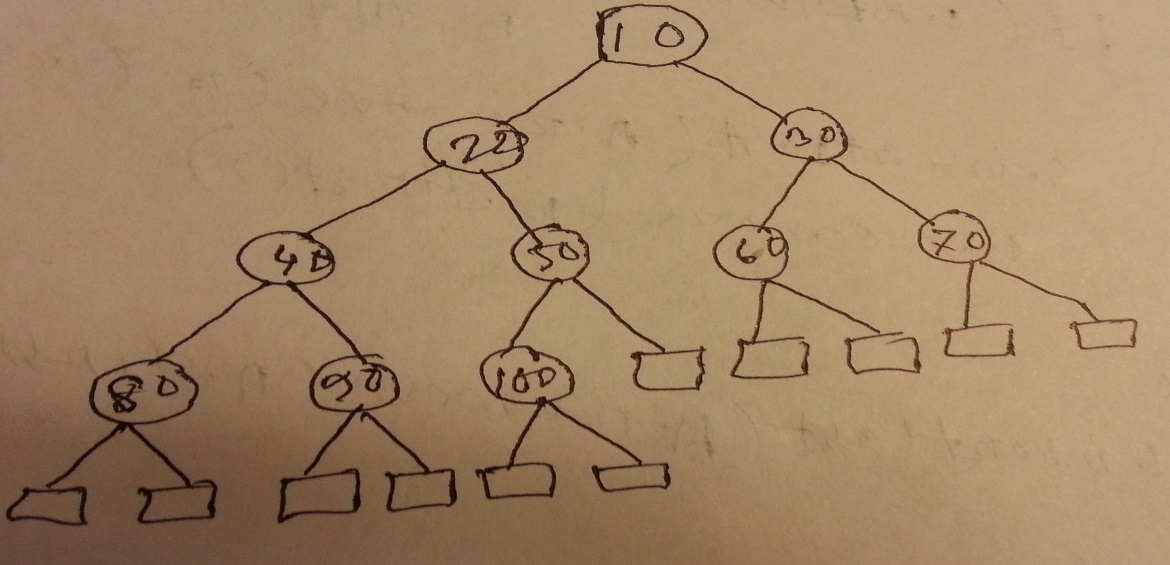
Yes, it is.

**Justification:**

Vector, S with 10 items in ascending order.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |

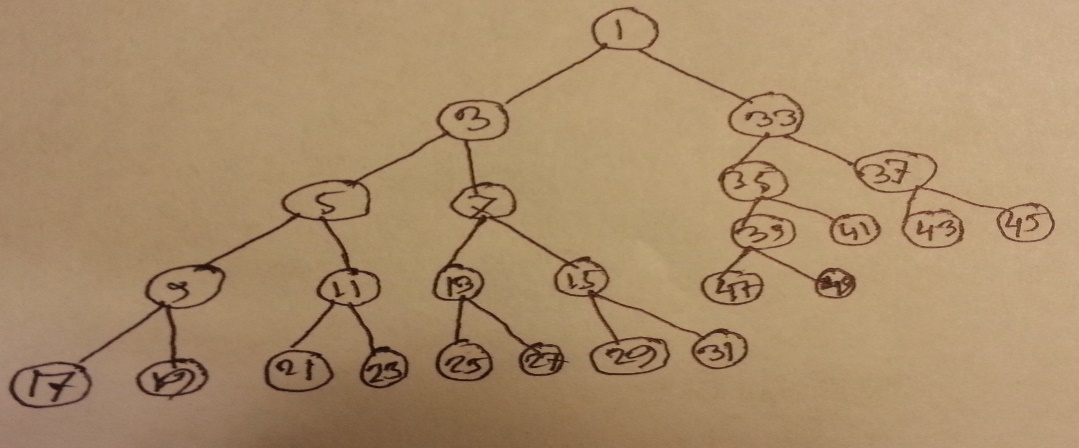
If we draw a tree, it will be like this.



As we know that, for heap

1. For all nodes except root key(child) >= key(parent).This tree maintain this property.
2. Root has minimum value. Here, minimum value 10 is in the root.
3. Subtrees are also heap and values are in ascending order from root to leaf

R-2-18 **Answer:**

****

C-2.32.**Answer:**

|  |
| --- |
| Algorithm getSmallerEqualKey**(**T**,**key**,**i**)**  Input**:** Vecotr T representing a Heap**,**index i of an element in the heap and a query key x  Output**:** Vecotr V contining all the keys which are less than or equal to given key  **if** i **<** T**.**size**()** **^** T**.**elementAtRank**(**i**)** **<=** key  V**.**insertLast**(**T**.**elementAtRank**(**i**))**  getSmallerEqualKey**(**T**,**key**,**2 **\*** i**)**  getSmallerEqualKey**(**T**,**key**,**2 **\*** i **+** 1**)**  **return** |