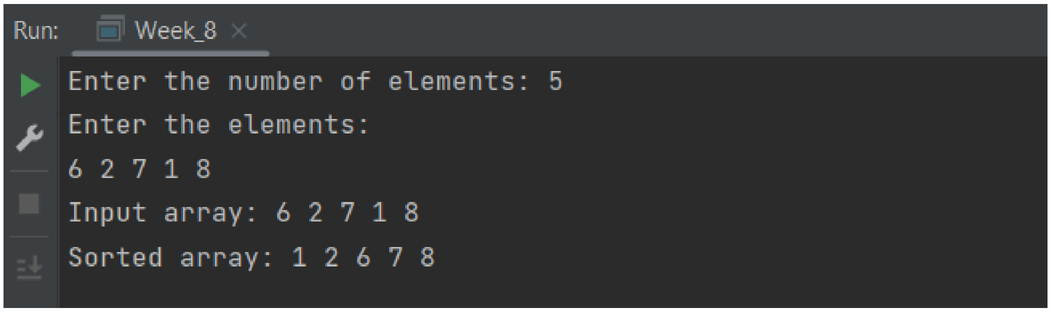
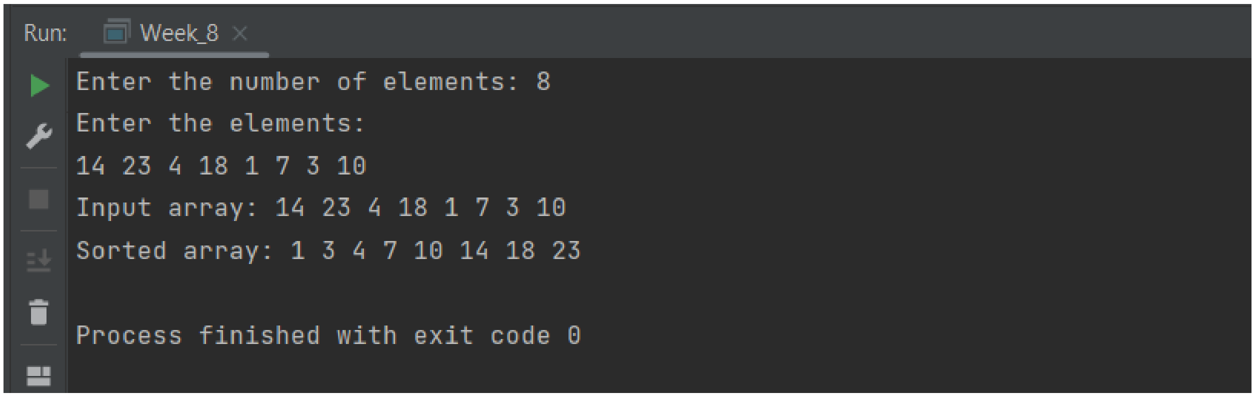
Github link -: https://github.com/StupidME2000/Sem-4-DSA-ICA-Week-8-Q1-Heap-Data-Structure.git

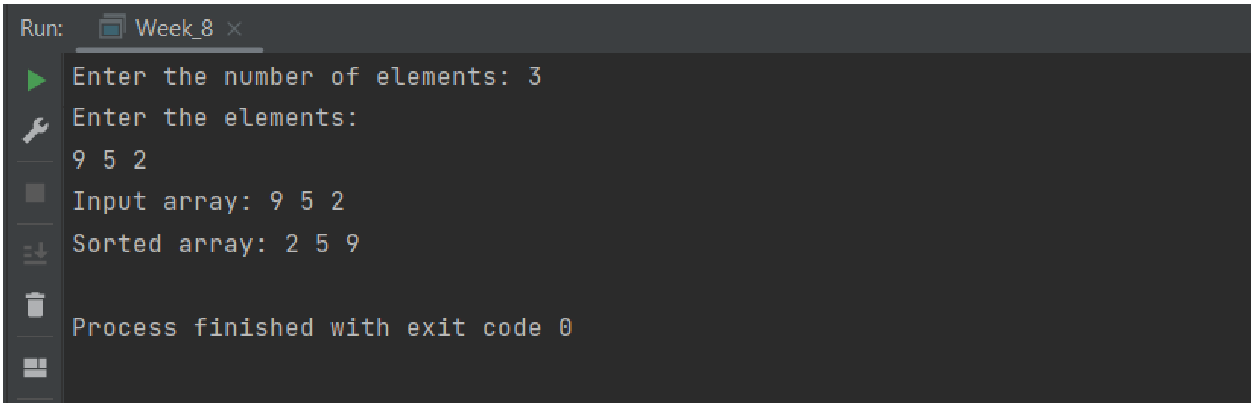


```
Enter the number of elements: 5
Enter the elements:
6 2 7 1 8
Input array: 6 2 7 1 8
Sorted array: 1 2 6 7 8
```



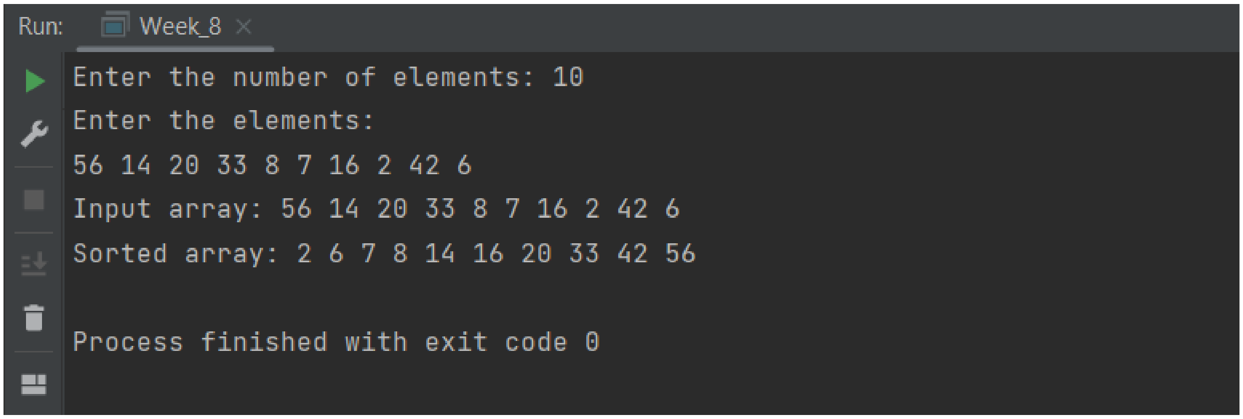
```
Enter the number of elements: 8
Enter the elements:
14 23 4 18 1 7 3 10
Input array: 14 23 4 18 1 7 3 10
Sorted array: 1 3 4 7 10 14 18 23

Process finished with exit code 0
```



```
Enter the number of elements: 3
Enter the elements:
9 5 2
Input array: 9 5 2
Sorted array: 2 5 9

Process finished with exit code 0
```



```
Enter the number of elements: 10
Enter the elements:
56 14 20 33 8 7 16 2 42 6
Input array: 56 14 20 33 8 7 16 2 42 6
Sorted array: 2 6 7 8 14 16 20 33 42 56

Process finished with exit code 0
```

Heap Sort

The time complexity of Heap Sort is O(nlogn) in the worst, average, and best case. This makes it an efficient sorting algorithm, and it is widely used in practice.

The algorithm consists of two main steps: building a heap and then extracting elements from the heap. The first step involves building a heap from the input array, which takes O(n) time. The second step involves extracting the maximum element from the heap n times, which takes O(nlogn) time. Therefore, the total time complexity of Heap Sort is O(n) + O(nlogn) = O(nlogn).

Heap Sort has a better worst-case time complexity than many other comparison-based sorting algorithms, such as Quick Sort and Merge Sort, which have a worst-case time complexity of O(n^2) and O(nlogn), respectively. However, Heap Sort has a higher constant factor than these algorithms, making it slower for small input sizes. Additionally, Heap Sort is not a stable sorting algorithm, which means that it does not preserve the relative order of equal elements in the input array.