# GTU Department of Computer Engineering
# CSE 222/505 - Spring 2020
# Homework 6 Report
# Q1

Buğra Eren Yılmaz
1801042669

# Shell Sort A

```
Arr: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Gap: 5
Pick arr[5]: 6
Shifting gap-sorted elements.
Put last picked arr[5]: 6 to arr[5]
arr[5]: 6 is correct
-----------------------
Pick arr[6]: 7
Shifting gap-sorted elements.
Put last picked arr[6]: 7 to arr[6]
arr[6]: 7 is correct
-----------------------
Pick arr[7]: 8
Shifting gap-sorted elements.
Put last picked arr[7]: 8 to arr[7]
arr[7]: 8 is correct
-----------------------
Pick arr[8]: 9
Shifting gap-sorted elements.
Put last picked arr[8]: 9 to arr[8]
arr[8]: 9 is correct
-----------------------
Pick arr[9]: 10
Shifting gap-sorted elements.
Put last picked arr[9]: 10 to arr[9]
arr[9]: 10 is correct
-----------------------
Reduce gap.
Arr: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Gap: 2
Pick arr[2]: 3
Shifting gap-sorted elements.
Put last picked arr[2]: 3 to arr[2]
arr[2]: 3 is correct
-----------------------
Pick arr[3]: 4
Shifting gap-sorted elements.
Put last picked arr[3]: 4 to arr[3]
arr[3]: 4 is correct
-----------------------
Pick arr[4]: 5
Shifting gap-sorted elements.
Put last picked arr[4]: 5 to arr[4]
arr[4]: 5 is correct
-----------------------
Pick arr[5]: 6
Shifting gap-sorted elements.
Put last picked arr[5]: 6 to arr[5]
arr[5]: 6 is correct
```

```
-----------------------
Pick arr[6]: 7
Shifting gap-sorted elements.
Put last picked arr[6]: 7 to arr[6]
arr[6]: 7 is correct
-----------------------
Pick arr[7]: 8
Shifting gap-sorted elements.
Put last picked arr[7]: 8 to arr[7]
arr[7]: 8 is correct
-----------------------
Pick arr[8]: 9
Shifting gap-sorted elements.
Put last picked arr[8]: 9 to arr[8]
arr[8]: 9 is correct
-----------------------
Pick arr[9]: 10
Shifting gap-sorted elements.
Put last picked arr[9]: 10 to arr[9]
arr[9]: 10 is correct
-----------------------
Reduce gap.
Arr: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Gap: 1
Pick arr[1]: 2
Shifting gap-sorted elements.
Put last picked arr[1]: 2 to arr[1]
arr[1]: 2 is correct
-----------------------
Pick arr[2]: 3
Shifting gap-sorted elements.
Put last picked arr[2]: 3 to arr[2]
arr[2]: 3 is correct
-----------------------
Pick arr[3]: 4
Shifting gap-sorted elements.
Put last picked arr[3]: 4 to arr[3]
arr[3]: 4 is correct
-----------------------
Pick arr[4]: 5
Shifting gap-sorted elements.
Put last picked arr[4]: 5 to arr[4]
arr[4]: 5 is correct
-----------------------
Pick arr[5]: 6
Shifting gap-sorted elements.
Put last picked arr[5]: 6 to arr[5]
arr[5]: 6 is correct
-----------------------
Pick arr[6]: 7
Shifting gap-sorted elements.
Put last picked arr[6]: 7 to arr[6]
arr[6]: 7 is correct
```

```
-----------------------
Pick arr[7]: 8
Shifting gap-sorted elements.
Put last picked arr[7]: 8 to arr[7]
arr[7]: 8 is correct
-----------------------
Pick arr[8]: 9
Shifting gap-sorted elements.
Put last picked arr[8]: 9 to arr[8]
arr[8]: 9 is correct
-----------------------
Pick arr[9]: 10
Shifting gap-sorted elements.
Put last picked arr[9]: 10 to arr[9]
arr[9]: 10 is correct
-----------------------
Reduce gap.
Arr: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Gap: 0
Shell Sort done
Comparisons count: 22
Displacement count: 0
```

# Shell Sort B

```
Arr: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
Gap: 5
Pick arr[5]: 5
Shifting gap-sorted elements.
Swap 5 with 10
Put last picked arr[5]: 10 to arr[0]
arr[5]: 5 is correct
-----------------------
Pick arr[6]: 4
Shifting gap-sorted elements.
Swap 4 with 9
Put last picked arr[6]: 9 to arr[1]
arr[6]: 4 is correct
-----------------------
Pick arr[7]: 3
Shifting gap-sorted elements.
Swap 3 with 8
Put last picked arr[7]: 8 to arr[2]
arr[7]: 3 is correct
-----------------------
Pick arr[8]: 2
Shifting gap-sorted elements.
Swap 2 with 7
Put last picked arr[8]: 7 to arr[3]
arr[8]: 2 is correct
-----------------------
Pick arr[9]: 1
Shifting gap-sorted elements.
```

```
Swap 1 with 6
Put last picked arr[9]: 6 to arr[4]
arr[9]: 1 is correct
------------------------
Reduce gap.
Arr: [5, 4, 3, 2, 1, 10, 9, 8, 7, 6]
Gap: 2
Pick arr[2]: 3
Shifting gap-sorted elements.
Swap 3 with 5
Put last picked arr[2]: 5 to arr[0]
arr[2]: 3 is correct
------------------------
Pick arr[3]: 2
Shifting gap-sorted elements.
Swap 2 with 4
Put last picked arr[3]: 4 to arr[1]
arr[3]: 2 is correct
------------------------
Pick arr[4]: 1
Shifting gap-sorted elements.
Swap 1 with 5
Swap 5 with 3
Put last picked arr[4]: 5 to arr[0]
arr[4]: 1 is correct
------------------------
Pick arr[5]: 10
Shifting gap-sorted elements.
Put last picked arr[5]: 10 to arr[5]
arr[5]: 10 is correct
------------------------
Pick arr[6]: 9
Shifting gap-sorted elements.
Put last picked arr[6]: 9 to arr[6]
arr[6]: 9 is correct
------------------------
Pick arr[7]: 8
Shifting gap-sorted elements.
Swap 8 with 10
Put last picked arr[7]: 10 to arr[5]
arr[7]: 8 is correct
------------------------
Pick arr[8]: 7
Shifting gap-sorted elements.
Swap 7 with 9
Put last picked arr[8]: 9 to arr[6]
arr[8]: 7 is correct
------------------------
Pick arr[9]: 6
Shifting gap-sorted elements.
Swap 6 with 10
Swap 10 with 8
Put last picked arr[9]: 10 to arr[5]
```

```
arr[9]: 6 is correct
------------------------
Reduce gap.
Arr: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Gap: 1
Pick arr[1]: 2
Shifting gap-sorted elements.
Put last picked arr[1]: 2 to arr[1]
arr[1]: 2 is correct
------------------------
Pick arr[2]: 3
Shifting gap-sorted elements.
Put last picked arr[2]: 3 to arr[2]
arr[2]: 3 is correct
------------------------
Pick arr[3]: 4
Shifting gap-sorted elements.
Put last picked arr[3]: 4 to arr[3]
arr[3]: 4 is correct
------------------------
Pick arr[4]: 5
Shifting gap-sorted elements.
Put last picked arr[4]: 5 to arr[4]
arr[4]: 5 is correct
------------------------
Pick arr[5]: 6
Shifting gap-sorted elements.
Put last picked arr[5]: 6 to arr[5]
arr[5]: 6 is correct
------------------------
Pick arr[6]: 7
Shifting gap-sorted elements.
Put last picked arr[6]: 7 to arr[6]
arr[6]: 7 is correct
------------------------
Pick arr[7]: 8
Shifting gap-sorted elements.
Put last picked arr[7]: 8 to arr[7]
arr[7]: 8 is correct
------------------------
Pick arr[8]: 9
Shifting gap-sorted elements.
Put last picked arr[8]: 9 to arr[8]
arr[8]: 9 is correct
------------------------
Pick arr[9]: 10
Shifting gap-sorted elements.
Put last picked arr[9]: 10 to arr[9]
arr[9]: 10 is correct
------------------------
Reduce gap.
Arr: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Gap: 0
```

```
Shell Sort done
Comparisons count: 35
Displacement count: 24
```

## Shell Sort C

```
Arr: [5, 2, 13, 9, 1, 7, 6, 8, 1, 15, 4, 11]
Gap: 6
Pick arr[6]: 6
Shifting gap-sorted elements.
Put last picked arr[6]: 6 to arr[6]
arr[6]: 6 is correct
------------------------
Pick arr[7]: 8
Shifting gap-sorted elements.
Put last picked arr[7]: 8 to arr[7]
arr[7]: 8 is correct
------------------------
Pick arr[8]: 1
Shifting gap-sorted elements.
Swap 1 with 13
Put last picked arr[8]: 13 to arr[2]
arr[8]: 1 is correct
------------------------
Pick arr[9]: 15
Shifting gap-sorted elements.
Put last picked arr[9]: 15 to arr[9]
arr[9]: 15 is correct
------------------------
Pick arr[10]: 4
Shifting gap-sorted elements.
Put last picked arr[10]: 4 to arr[10]
arr[10]: 4 is correct
------------------------
Pick arr[11]: 11
Shifting gap-sorted elements.
Put last picked arr[11]: 11 to arr[11]
arr[11]: 11 is correct
------------------------
Reduce gap.
Arr: [5, 2, 1, 9, 1, 7, 6, 8, 13, 15, 4, 11]
Gap: 3
Pick arr[3]: 9
Shifting gap-sorted elements.
Put last picked arr[3]: 9 to arr[3]
arr[3]: 9 is correct
------------------------
Pick arr[4]: 1
```

```
Shifting gap-sorted elements.
Swap 1 with 2
Put last picked arr[4]: 2 to arr[1]
arr[4]: 1 is correct
------------------------
Pick arr[5]: 7
Shifting gap-sorted elements.
Put last picked arr[5]: 7 to arr[5]
arr[5]: 7 is correct
------------------------
Pick arr[6]: 6
Shifting gap-sorted elements.
Swap 6 with 9
Put last picked arr[6]: 9 to arr[3]
arr[6]: 6 is correct
------------------------
Pick arr[7]: 8
Shifting gap-sorted elements.
Put last picked arr[7]: 8 to arr[7]
arr[7]: 8 is correct
------------------------
Pick arr[8]: 13
Shifting gap-sorted elements.
Put last picked arr[8]: 13 to arr[8]
arr[8]: 13 is correct
------------------------
Pick arr[9]: 15
Shifting gap-sorted elements.
Put last picked arr[9]: 15 to arr[9]
arr[9]: 15 is correct
------------------------
Pick arr[10]: 4
Shifting gap-sorted elements.
Swap 4 with 8
Put last picked arr[10]: 8 to arr[7]
arr[10]: 4 is correct
------------------------
Pick arr[11]: 11
Shifting gap-sorted elements.
Swap 11 with 13
Put last picked arr[11]: 13 to arr[8]
arr[11]: 11 is correct
------------------------
Reduce gap.
Arr: [5, 1, 1, 6, 2, 7, 9, 4, 11, 15, 8, 13]
Gap: 1
Pick arr[1]: 1
Shifting gap-sorted elements.
```

```
Swap 1 with 5
Put last picked arr[1]: 5 to arr[0]
arr[1]: 1 is correct
------------------------
Pick arr[2]: 1
Shifting gap-sorted elements.
Swap 1 with 5
Put last picked arr[2]: 5 to arr[1]
arr[2]: 1 is correct
------------------------
Pick arr[3]: 6
Shifting gap-sorted elements.
Put last picked arr[3]: 6 to arr[3]
arr[3]: 6 is correct
------------------------
Pick arr[4]: 2
Shifting gap-sorted elements.
Swap 2 with 6
Swap 6 with 5
Put last picked arr[4]: 6 to arr[2]
arr[4]: 2 is correct
------------------------
Pick arr[5]: 7
Shifting gap-sorted elements.
Put last picked arr[5]: 7 to arr[5]
arr[5]: 7 is correct
------------------------
Pick arr[6]: 9
Shifting gap-sorted elements.
Put last picked arr[6]: 9 to arr[6]
arr[6]: 9 is correct
------------------------
Pick arr[7]: 4
Shifting gap-sorted elements.
Swap 4 with 9
Swap 9 with 7
Swap 7 with 6
Swap 6 with 5
Put last picked arr[7]: 9 to arr[3]
arr[7]: 4 is correct
------------------------
Pick arr[8]: 11
Shifting gap-sorted elements.
Put last picked arr[8]: 11 to arr[8]
arr[8]: 11 is correct
------------------------
Pick arr[9]: 15
Shifting gap-sorted elements.
```

```
Put last picked arr[9]: 15 to arr[9]
arr[9]: 15 is correct
------------------------
Pick arr[10]: 8
Shifting gap-sorted elements.
Swap 8 with 15
Swap 15 with 11
Swap 11 with 9
Put last picked arr[10]: 15 to arr[7]
arr[10]: 8 is correct
------------------------
Pick arr[11]: 13
Shifting gap-sorted elements.
Swap 13 with 15
Put last picked arr[11]: 15 to arr[10]
arr[11]: 13 is correct
------------------------
Reduce gap.
Arr: [1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15]
Gap: 0
Shell Sort done
Comparisons count: 43
Displacement count: 28
```

Shell Sort D

```
Arr: ['S', 'B', 'I', 'M', 'H', 'Q', 'C', 'L', 'R', 'E', 'P', 'K']
Gap: 6
Pick arr[6]: C
Shifting gap-sorted elements.
Swap C with S
Put last picked arr[6]: S to arr[0]
arr[6]: C is correct
------------------------
Pick arr[7]: L
Shifting gap-sorted elements.
Put last picked arr[7]: L to arr[7]
arr[7]: L is correct
------------------------
Pick arr[8]: R
Shifting gap-sorted elements.
Put last picked arr[8]: R to arr[8]
arr[8]: R is correct
------------------------
Pick arr[9]: E
Shifting gap-sorted elements.
Swap E with M
Put last picked arr[9]: M to arr[3]
arr[9]: E is correct
```

```
------------------------
Pick arr[10]: P
Shifting gap-sorted elements.
Put last picked arr[10]: P to arr[10]
arr[10]: P is correct
------------------------
Pick arr[11]: K
Shifting gap-sorted elements.
Swap K with Q
Put last picked arr[11]: Q to arr[5]
arr[11]: K is correct
------------------------
Reduce gap.
Arr: ['C', 'B', 'I', 'E', 'H', 'K', 'S', 'L', 'R', 'M', 'P', 'Q']
Gap: 3
Pick arr[3]: E
Shifting gap-sorted elements.
Put last picked arr[3]: E to arr[3]
arr[3]: E is correct
------------------------
Pick arr[4]: H
Shifting gap-sorted elements.
Put last picked arr[4]: H to arr[4]
arr[4]: H is correct
------------------------
Pick arr[5]: K
Shifting gap-sorted elements.
Put last picked arr[5]: K to arr[5]
arr[5]: K is correct
------------------------
Pick arr[6]: S
Shifting gap-sorted elements.
Put last picked arr[6]: S to arr[6]
arr[6]: S is correct
------------------------
Pick arr[7]: L
Shifting gap-sorted elements.
Put last picked arr[7]: L to arr[7]
arr[7]: L is correct
------------------------
Pick arr[8]: R
Shifting gap-sorted elements.
Put last picked arr[8]: R to arr[8]
arr[8]: R is correct
------------------------
Pick arr[9]: M
Shifting gap-sorted elements.
Swap M with S
```

```
Put last picked arr[9]: S to arr[6]
arr[9]: M is correct
------------------------
Pick arr[10]: P
Shifting gap-sorted elements.
Put last picked arr[10]: P to arr[10]
arr[10]: P is correct
------------------------
Pick arr[11]: Q
Shifting gap-sorted elements.
Swap Q with R
Put last picked arr[11]: R to arr[8]
arr[11]: Q is correct
------------------------
Reduce gap.
Arr: ['C', 'B', 'I', 'E', 'H', 'K', 'M', 'L', 'Q', 'S', 'P', 'R']
Gap: 1
Pick arr[1]: B
Shifting gap-sorted elements.
Swap B with C
Put last picked arr[1]: C to arr[0]
arr[1]: B is correct
------------------------
Pick arr[2]: I
Shifting gap-sorted elements.
Put last picked arr[2]: I to arr[2]
arr[2]: I is correct
------------------------
Pick arr[3]: E
Shifting gap-sorted elements.
Swap E with I
Put last picked arr[3]: I to arr[2]
arr[3]: E is correct
------------------------
Pick arr[4]: H
Shifting gap-sorted elements.
Swap H with I
Put last picked arr[4]: I to arr[3]
arr[4]: H is correct
------------------------
Pick arr[5]: K
Shifting gap-sorted elements.
Put last picked arr[5]: K to arr[5]
arr[5]: K is correct
------------------------
Pick arr[6]: M
Shifting gap-sorted elements.
Put last picked arr[6]: M to arr[6]
```

```
arr[6]: M is correct
------------------------
Pick arr[7]: L
Shifting gap-sorted elements.
Swap L with M
Put last picked arr[7]: M to arr[6]
arr[7]: L is correct
------------------------
Pick arr[8]: Q
Shifting gap-sorted elements.
Put last picked arr[8]: Q to arr[8]
arr[8]: Q is correct
------------------------
Pick arr[9]: S
Shifting gap-sorted elements.
Put last picked arr[9]: S to arr[9]
arr[9]: S is correct
------------------------
Pick arr[10]: P
Shifting gap-sorted elements.
Swap P with S
Swap S with Q
Put last picked arr[10]: S to arr[8]
arr[10]: P is correct
------------------------
Pick arr[11]: R
Shifting gap-sorted elements.
Swap R with S
Put last picked arr[11]: S to arr[10]
arr[11]: R is correct
------------------------
Reduce gap.
Arr: ['B', 'C', 'E', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
Gap: 0
Shell Sort done
Comparisons count: 38
Displacement count: 23
```

## Merge Sort A

```
Merge-Sorting: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Pick middle index: 5
Divide left half: [1, 2, 3, 4, 5]
Divide right half: [1, 2, 3, 4, 5]
Merge-Sort Left Half
Merge-Sorting: [1, 2, 3, 4, 5]
Pick middle index: 2
Divide left half: [1, 2]
Divide right half: [1, 2]
```

```
Merge-Sort Left Half
Merge-Sorting: [1, 2]
Pick middle index: 1
Divide left half: [1]
Divide right half: [1]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
It is sorted
------------
Combine left and right half
Combined halves: [1, 2]
Merge-Sort Right Half
Merge-Sorting: [3, 4, 5]
Pick middle index: 1
Divide left half: [3]
Divide right half: [3]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
Merge-Sorting: [4, 5]
Pick middle index: 1
Divide left half: [4]
Divide right half: [4]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
It is sorted
------------
Combine left and right half
Combined halves: [4, 5]
Combine left and right half
Combined halves: [3, 4, 5]
Combine left and right half
Combined halves: [1, 2, 3, 4, 5]
Merge-Sort Right Half
Merge-Sorting: [6, 7, 8, 9, 10]
Pick middle index: 2
Divide left half: [6, 7]
Divide right half: [6, 7]
Merge-Sort Left Half
Merge-Sorting: [6, 7]
Pick middle index: 1
Divide left half: [6]
Divide right half: [6]
```

```
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
It is sorted
------------
Combine left and right half
Combined halves: [6, 7]
Merge-Sort Right Half
Merge-Sorting: [8, 9, 10]
Pick middle index: 1
Divide left half: [8]
Divide right half: [8]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
Merge-Sorting: [9, 10]
Pick middle index: 1
Divide left half: [9]
Divide right half: [9]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
It is sorted
------------
Combine left and right half
Combined halves: [9, 10]
Combine left and right half
Combined halves: [8, 9, 10]
Combine left and right half
Combined halves: [6, 7, 8, 9, 10]
Combine left and right half
Combined halves: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Merge sort done.
Comparisons count: 15
Displacement count: 0
```

## Merge Sort B

```
Merge-Sorting: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
Pick middle index: 5
Divide left half: [10, 9, 8, 7, 6]
Divide right half: [10, 9, 8, 7, 6]
Merge-Sort Left Half
Merge-Sorting: [10, 9, 8, 7, 6]
Pick middle index: 2
```

```
Divide left half: [10, 9]
Divide right half: [10, 9]
Merge-Sort Left Half
Merge-Sorting: [10, 9]
Pick middle index: 1
Divide left half: [10]
Divide right half: [10]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
It is sorted
------------
Combine left and right half
Combined halves: [9, 10]
Merge-Sort Right Half
Merge-Sorting: [8, 7, 6]
Pick middle index: 1
Divide left half: [8]
Divide right half: [8]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
Merge-Sorting: [7, 6]
Pick middle index: 1
Divide left half: [7]
Divide right half: [7]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
It is sorted
------------
Combine left and right half
Combined halves: [6, 7]
Combine left and right half
Combined halves: [6, 7, 8]
Combine left and right half
Combined halves: [6, 7, 8, 9, 10]
Merge-Sort Right Half
Merge-Sorting: [5, 4, 3, 2, 1]
Pick middle index: 2
Divide left half: [5, 4]
Divide right half: [5, 4]
Merge-Sort Left Half
Merge-Sorting: [5, 4]
Pick middle index: 1
```

```
Divide left half: [5]
Divide right half: [5]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
It is sorted
------------
Combine left and right half
Combined halves: [4, 5]
Merge-Sort Right Half
Merge-Sorting: [3, 2, 1]
Pick middle index: 1
Divide left half: [3]
Divide right half: [3]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
Merge-Sorting: [2, 1]
Pick middle index: 1
Divide left half: [2]
Divide right half: [2]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
It is sorted
------------
Combine left and right half
Combined halves: [1, 2]
Combine left and right half
Combined halves: [1, 2, 3]
Combine left and right half
Combined halves: [1, 2, 3, 4, 5]
Combine left and right half
Combined halves: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Merge sort done.
Comparisons count: 19
Displacement count: 30
```

## Merge Sort C

```
Merge-Sorting: [5, 2, 13, 9, 1, 7, 6, 8, 1, 15, 4, 11]
Pick middle index: 6
Divide left half: [5, 2, 13, 9, 1, 7]
Divide right half: [5, 2, 13, 9, 1, 7]
Merge-Sort Left Half
```

```
Merge-Sorting: [5, 2, 13, 9, 1, 7]
Pick middle index: 3
Divide left half: [5, 2, 13]
Divide right half: [5, 2, 13]
Merge-Sort Left Half
Merge-Sorting: [5, 2, 13]
Pick middle index: 1
Divide left half: [5]
Divide right half: [5]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
Merge-Sorting: [2, 13]
Pick middle index: 1
Divide left half: [2]
Divide right half: [2]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
It is sorted
------------
Combine left and right half
Combined halves: [2, 13]
Combine left and right half
Combined halves: [2, 5, 13]
Merge-Sort Right Half
Merge-Sorting: [9, 1, 7]
Pick middle index: 1
Divide left half: [9]
Divide right half: [9]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
Merge-Sorting: [1, 7]
Pick middle index: 1
Divide left half: [1]
Divide right half: [1]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
It is sorted
------------
Combine left and right half
Combined halves: [1, 7]
```

```
Combine left and right half
Combined halves: [1, 7, 9]
Combine left and right half
Combined halves: [1, 2, 5, 7, 9, 13]
Merge-Sort Right Half
Merge-Sorting: [6, 8, 1, 15, 4, 11]
Pick middle index: 3
Divide left half: [6, 8, 1]
Divide right half: [6, 8, 1]
Merge-Sort Left Half
Merge-Sorting: [6, 8, 1]
Pick middle index: 1
Divide left half: [6]
Divide right half: [6]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
Merge-Sorting: [8, 1]
Pick middle index: 1
Divide left half: [8]
Divide right half: [8]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
It is sorted
------------
Combine left and right half
Combined halves: [1, 8]
Combine left and right half
Combined halves: [1, 6, 8]
Merge-Sort Right Half
Merge-Sorting: [15, 4, 11]
Pick middle index: 1
Divide left half: [15]
Divide right half: [15]
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
Merge-Sorting: [4, 11]
Pick middle index: 1
Divide left half: [4]
Divide right half: [4]
Merge-Sort Left Half
It is sorted
------------
```

```
Merge-Sort Right Half
It is sorted
------------
Combine left and right half
Combined halves: [4, 11]
Combine left and right half
Combined halves: [4, 11, 15]
Combine left and right half
Combined halves: [1, 4, 6, 8, 11, 15]
Combine left and right half
Combined halves: [1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15]
Merge sort done.
Comparisons count: 32
Displacement count: 31
```

## Merge Sort D

```
Merge-Sorting: ['S', 'B', 'I', 'M', 'H', 'Q', 'C', 'L', 'R', 'E', 'P', 'K']
Pick middle index: 6
Divide left half: ['S', 'B', 'I', 'M', 'H', 'Q']
Divide right half: ['S', 'B', 'I', 'M', 'H', 'Q']
Merge-Sort Left Half
Merge-Sorting: ['S', 'B', 'I', 'M', 'H', 'Q']
Pick middle index: 3
Divide left half: ['S', 'B', 'I']
Divide right half: ['S', 'B', 'I']
Merge-Sort Left Half
Merge-Sorting: ['S', 'B', 'I']
Pick middle index: 1
Divide left half: ['S']
Divide right half: ['S']
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
Merge-Sorting: ['B', 'I']
Pick middle index: 1
Divide left half: ['B']
Divide right half: ['B']
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
It is sorted
------------
Combine left and right half
Combined halves: ['B', 'I']
Combine left and right half
Combined halves: ['B', 'I', 'S']
```

```
Merge-Sort Right Half
Merge-Sorting: ['M', 'H', 'Q']
Pick middle index: 1
Divide left half: ['M']
Divide right half: ['M']
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
Merge-Sorting: ['H', 'Q']
Pick middle index: 1
Divide left half: ['H']
Divide right half: ['H']
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
It is sorted
------------
Combine left and right half
Combined halves: ['H', 'Q']
Combine left and right half
Combined halves: ['H', 'M', 'Q']
Combine left and right half
Combined halves: ['B', 'H', 'I', 'M', 'Q', 'S']
Merge-Sort Right Half
Merge-Sorting: ['C', 'L', 'R', 'E', 'P', 'K']
Pick middle index: 3
Divide left half: ['C', 'L', 'R']
Divide right half: ['C', 'L', 'R']
Merge-Sort Left Half
Merge-Sorting: ['C', 'L', 'R']
Pick middle index: 1
Divide left half: ['C']
Divide right half: ['C']
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
Merge-Sorting: ['L', 'R']
Pick middle index: 1
Divide left half: ['L']
Divide right half: ['L']
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
It is sorted
```

```
------------
Combine left and right half
Combined halves: ['L', 'R']
Combine left and right half
Combined halves: ['C', 'L', 'R']
Merge-Sort Right Half
Merge-Sorting: ['E', 'P', 'K']
Pick middle index: 1
Divide left half: ['E']
Divide right half: ['E']
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
Merge-Sorting: ['P', 'K']
Pick middle index: 1
Divide left half: ['P']
Divide right half: ['P']
Merge-Sort Left Half
It is sorted
------------
Merge-Sort Right Half
It is sorted
------------
Combine left and right half
Combined halves: ['K', 'P']
Combine left and right half
Combined halves: ['E', 'K', 'P']
Combine left and right half
Combined halves: ['C', 'E', 'K', 'L', 'P', 'R']
Combine left and right half
Combined halves: ['B', 'C', 'E', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
Merge sort done.
Comparisons count: 31
Displacement count: 27
```

## Heap Sort A

```
Heap sorting: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Building the maxheap
Heapify: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Heapified: [1, 2, 3, 4, 10, 6, 7, 8, 9, 5]
------------
Heapify: [1, 2, 3, 4, 10, 6, 7, 8, 9, 5]
Heapified: [1, 2, 3, 9, 10, 6, 7, 8, 4, 5]
------------
Heapify: [1, 2, 3, 9, 10, 6, 7, 8, 4, 5]
Heapified: [1, 2, 7, 9, 10, 6, 3, 8, 4, 5]
```

```
------------
Heapify: [1, 2, 7, 9, 10, 6, 3, 8, 4, 5]
Heapified: [1, 10, 7, 9, 5, 6, 3, 8, 4, 2]
------------
Heapify: [1, 10, 7, 9, 5, 6, 3, 8, 4, 2]
Heapified: [10, 9, 7, 8, 5, 6, 3, 1, 4, 2]
------------
Built max heap: [10, 9, 7, 8, 5, 6, 3, 1, 4, 2]
Extract elements one by one
Swap 2 with 10
Heapify: [2, 9, 7, 8, 5, 6, 3, 1, 4, 10]
Heapified: [9, 8, 7, 4, 5, 6, 3, 1, 2, 10]
Swap 2 with 9
Heapify: [2, 8, 7, 4, 5, 6, 3, 1, 9, 10]
Heapified: [8, 5, 7, 4, 2, 6, 3, 1, 9, 10]
Swap 1 with 8
Heapify: [1, 5, 7, 4, 2, 6, 3, 8, 9, 10]
Heapified: [7, 5, 6, 4, 2, 1, 3, 8, 9, 10]
Swap 3 with 7
Heapify: [3, 5, 6, 4, 2, 1, 7, 8, 9, 10]
Heapified: [6, 5, 3, 4, 2, 1, 7, 8, 9, 10]
Swap 1 with 6
Heapify: [1, 5, 3, 4, 2, 6, 7, 8, 9, 10]
Heapified: [5, 4, 3, 1, 2, 6, 7, 8, 9, 10]
Swap 2 with 5
Heapify: [2, 4, 3, 1, 5, 6, 7, 8, 9, 10]
Heapified: [4, 2, 3, 1, 5, 6, 7, 8, 9, 10]
Swap 1 with 4
Heapify: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Heapified: [3, 2, 1, 4, 5, 6, 7, 8, 9, 10]
Swap 1 with 3
Heapify: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Heapified: [2, 1, 3, 4, 5, 6, 7, 8, 9, 10]
Swap 1 with 2
Heapify: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Heapified: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Heap sort done: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Comparisons count: 105
Displacement count: 9
```

## Heap Sort B

```
Heap sorting: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
Building the maxheap
Heapify: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
Heapified: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
------------
Heapify: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
```

```
Heapified: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
------------
Heapify: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
Heapified: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
------------
Heapify: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
Heapified: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
------------
Heapify: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
Heapified: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
------------
Built max heap: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
Extract elements one by one
Swap 1 with 10
Heapify: [1, 9, 8, 7, 6, 5, 4, 3, 2, 10]
Heapified: [9, 7, 8, 3, 6, 5, 4, 1, 2, 10]
Swap 2 with 9
Heapify: [2, 7, 8, 3, 6, 5, 4, 1, 9, 10]
Heapified: [8, 7, 5, 3, 6, 2, 4, 1, 9, 10]
Swap 1 with 8
Heapify: [1, 7, 5, 3, 6, 2, 4, 8, 9, 10]
Heapified: [7, 6, 5, 3, 1, 2, 4, 8, 9, 10]
Swap 4 with 7
Heapify: [4, 6, 5, 3, 1, 2, 7, 8, 9, 10]
Heapified: [6, 4, 5, 3, 1, 2, 7, 8, 9, 10]
Swap 2 with 6
Heapify: [2, 4, 5, 3, 1, 6, 7, 8, 9, 10]
Heapified: [5, 4, 2, 3, 1, 6, 7, 8, 9, 10]
Swap 1 with 5
Heapify: [1, 4, 2, 3, 5, 6, 7, 8, 9, 10]
Heapified: [4, 3, 2, 1, 5, 6, 7, 8, 9, 10]
Swap 1 with 4
Heapify: [1, 3, 2, 4, 5, 6, 7, 8, 9, 10]
Heapified: [3, 1, 2, 4, 5, 6, 7, 8, 9, 10]
Swap 2 with 3
Heapify: [2, 1, 3, 4, 5, 6, 7, 8, 9, 10]
Heapified: [2, 1, 3, 4, 5, 6, 7, 8, 9, 10]
Swap 1 with 2
Heapify: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Heapified: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Heap sort done: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Comparisons count: 78
Displacement count: 9
```

## Heap Sort C

```
Heap sorting: [5, 2, 13, 9, 1, 7, 6, 8, 1, 15, 4, 11]
Building the maxheap
```

```
Heapify: [5, 2, 13, 9, 1, 7, 6, 8, 1, 15, 4, 11]
Heapified: [5, 2, 13, 9, 1, 11, 6, 8, 1, 15, 4, 7]
------------
Heapify: [5, 2, 13, 9, 1, 11, 6, 8, 1, 15, 4, 7]
Heapified: [5, 2, 13, 9, 15, 11, 6, 8, 1, 1, 4, 7]
------------
Heapify: [5, 2, 13, 9, 15, 11, 6, 8, 1, 1, 4, 7]
Heapified: [5, 2, 13, 9, 15, 11, 6, 8, 1, 1, 4, 7]
------------
Heapify: [5, 2, 13, 9, 15, 11, 6, 8, 1, 1, 4, 7]
Heapified: [5, 2, 13, 9, 15, 11, 6, 8, 1, 1, 4, 7]
------------
Heapify: [5, 2, 13, 9, 15, 11, 6, 8, 1, 1, 4, 7]
Heapified: [5, 15, 13, 9, 4, 11, 6, 8, 1, 1, 2, 7]
------------
Heapify: [5, 15, 13, 9, 4, 11, 6, 8, 1, 1, 2, 7]
Heapified: [15, 9, 13, 8, 4, 11, 6, 5, 1, 1, 2, 7]
------------
Built max heap: [15, 9, 13, 8, 4, 11, 6, 5, 1, 1, 2, 7]
Extract elements one by one
Swap 7 with 15
Heapify: [7, 9, 13, 8, 4, 11, 6, 5, 1, 1, 2, 15]
Heapified: [13, 9, 11, 8, 4, 7, 6, 5, 1, 1, 2, 15]
Swap 2 with 13
Heapify: [2, 9, 11, 8, 4, 7, 6, 5, 1, 1, 13, 15]
Heapified: [11, 9, 7, 8, 4, 2, 6, 5, 1, 1, 13, 15]
Swap 1 with 11
Heapify: [1, 9, 7, 8, 4, 2, 6, 5, 1, 11, 13, 15]
Heapified: [9, 8, 7, 5, 4, 2, 6, 1, 1, 11, 13, 15]
Swap 1 with 9
Heapify: [1, 8, 7, 5, 4, 2, 6, 1, 9, 11, 13, 15]
Heapified: [8, 5, 7, 1, 4, 2, 6, 1, 9, 11, 13, 15]
Swap 1 with 8
Heapify: [1, 5, 7, 1, 4, 2, 6, 8, 9, 11, 13, 15]
Heapified: [7, 5, 6, 1, 4, 2, 1, 8, 9, 11, 13, 15]
Swap 1 with 7
Heapify: [1, 5, 6, 1, 4, 2, 7, 8, 9, 11, 13, 15]
Heapified: [6, 5, 2, 1, 4, 1, 7, 8, 9, 11, 13, 15]
Swap 1 with 6
Heapify: [1, 5, 2, 1, 4, 6, 7, 8, 9, 11, 13, 15]
Heapified: [5, 4, 2, 1, 1, 6, 7, 8, 9, 11, 13, 15]
Swap 1 with 5
Heapify: [1, 4, 2, 1, 5, 6, 7, 8, 9, 11, 13, 15]
Heapified: [4, 1, 2, 1, 5, 6, 7, 8, 9, 11, 13, 15]
Swap 1 with 4
Heapify: [1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15]
Heapified: [2, 1, 1, 4, 5, 6, 7, 8, 9, 11, 13, 15]
Swap 1 with 2
```

```
Heapify: [1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15]
Heapified: [1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15]
Swap 1 with 1
Heapify: [1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15]
Heapified: [1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15]
Heap sort done: [1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15]
Comparisons count: 123
Displacement count: 10
```

## Heap Sort D

```
Heap sorting: ['S', 'B', 'I', 'M', 'H', 'Q', 'C', 'L', 'R', 'E', 'P', 'K']
Building the maxheap
Heapify: ['S', 'B', 'I', 'M', 'H', 'Q', 'C', 'L', 'R', 'E', 'P', 'K']
Heapified: ['S', 'B', 'I', 'M', 'H', 'Q', 'C', 'L', 'R', 'E', 'P', 'K']
------------
Heapify: ['S', 'B', 'I', 'M', 'H', 'Q', 'C', 'L', 'R', 'E', 'P', 'K']
Heapified: ['S', 'B', 'I', 'M', 'P', 'Q', 'C', 'L', 'R', 'E', 'H', 'K']
------------
Heapify: ['S', 'B', 'I', 'M', 'P', 'Q', 'C', 'L', 'R', 'E', 'H', 'K']
Heapified: ['S', 'B', 'I', 'R', 'P', 'Q', 'C', 'L', 'M', 'E', 'H', 'K']
------------
Heapify: ['S', 'B', 'I', 'R', 'P', 'Q', 'C', 'L', 'M', 'E', 'H', 'K']
Heapified: ['S', 'B', 'Q', 'R', 'P', 'K', 'C', 'L', 'M', 'E', 'H', 'I']
------------
Heapify: ['S', 'B', 'Q', 'R', 'P', 'K', 'C', 'L', 'M', 'E', 'H', 'I']
Heapified: ['S', 'R', 'Q', 'M', 'P', 'K', 'C', 'L', 'B', 'E', 'H', 'I']
------------
Heapify: ['S', 'R', 'Q', 'M', 'P', 'K', 'C', 'L', 'B', 'E', 'H', 'I']
Heapified: ['S', 'R', 'Q', 'M', 'P', 'K', 'C', 'L', 'B', 'E', 'H', 'I']
------------
Built max heap: ['S', 'R', 'Q', 'M', 'P', 'K', 'C', 'L', 'B', 'E', 'H', 'I']
Extract elements one by one
Swap I with S
Heapify: ['I', 'R', 'Q', 'M', 'P', 'K', 'C', 'L', 'B', 'E', 'H', 'S']
Heapified: ['R', 'P', 'Q', 'M', 'I', 'K', 'C', 'L', 'B', 'E', 'H', 'S']
Swap H with R
Heapify: ['H', 'P', 'Q', 'M', 'I', 'K', 'C', 'L', 'B', 'E', 'R', 'S']
Heapified: ['Q', 'P', 'K', 'M', 'I', 'H', 'C', 'L', 'B', 'E', 'R', 'S']
Swap E with Q
Heapify: ['E', 'P', 'K', 'M', 'I', 'H', 'C', 'L', 'B', 'Q', 'R', 'S']
Heapified: ['P', 'M', 'K', 'L', 'I', 'H', 'C', 'E', 'B', 'Q', 'R', 'S']
Swap B with P
Heapify: ['B', 'M', 'K', 'L', 'I', 'H', 'C', 'E', 'P', 'Q', 'R', 'S']
Heapified: ['M', 'L', 'K', 'E', 'I', 'H', 'C', 'B', 'P', 'Q', 'R', 'S']
Swap B with M
Heapify: ['B', 'L', 'K', 'E', 'I', 'H', 'C', 'M', 'P', 'Q', 'R', 'S']
Heapified: ['L', 'I', 'K', 'E', 'B', 'H', 'C', 'M', 'P', 'Q', 'R', 'S']
```

```
Swap C with L
Heapify: ['C', 'I', 'K', 'E', 'B', 'H', 'L', 'M', 'P', 'Q', 'R', 'S']
Heapified: ['K', 'I', 'H', 'E', 'B', 'C', 'L', 'M', 'P', 'Q', 'R', 'S']
Swap C with K
Heapify: ['C', 'I', 'H', 'E', 'B', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
Heapified: ['I', 'E', 'H', 'C', 'B', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
Swap B with I
Heapify: ['B', 'E', 'H', 'C', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
Heapified: ['H', 'E', 'B', 'C', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
Swap C with H
Heapify: ['C', 'E', 'B', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
Heapified: ['E', 'C', 'B', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
Swap B with E
Heapify: ['B', 'C', 'E', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
Heapified: ['C', 'B', 'E', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
Swap B with C
Heapify: ['B', 'C', 'E', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
Heapified: ['B', 'C', 'E', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
Heap sort done: ['B', 'C', 'E', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
Comparisons count: 126
Displacement count: 11
```

## Quick Sort A

```
Quick sorting: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 0
end index: 9
Sorting part: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Pivot: 10
Small index: -1
Sorted part.
Part index: 9
Seperately quick sort parts
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 0
end index: 8
Sorting part: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Pivot: 9
Small index: -1
Sorted part.
Part index: 8
Seperately quick sort parts
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 0
end index: 7
Sorting part: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Pivot: 8
```

```
Small index: -1
Sorted part.
Part index: 7
Seperately quick sort parts
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 0
end index: 6
Sorting part: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Pivot: 7
Small index: -1
Sorted part.
Part index: 6
Seperately quick sort parts
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 0
end index: 5
Sorting part: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Pivot: 6
Small index: -1
Sorted part.
Part index: 5
Seperately quick sort parts
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 0
end index: 4
Sorting part: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Pivot: 5
Small index: -1
Sorted part.
Part index: 4
Seperately quick sort parts
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 0
end index: 3
Sorting part: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Pivot: 4
Small index: -1
Sorted part.
Part index: 3
Seperately quick sort parts
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 0
end index: 2
Sorting part: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Pivot: 3
Small index: -1
Sorted part.
Part index: 2
```

```
Seperately quick sort parts
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 0
end index: 1
Sorting part: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Pivot: 2
Small index: -1
Sorted part.
Part index: 1
Seperately quick sort parts
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 0
end index: 0
------------
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 2
end index: 1
------------
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 3
end index: 2
------------
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 4
end index: 3
------------
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 5
end index: 4
------------
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 6
end index: 5
------------
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 7
end index: 6
------------
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 8
end index: 7
------------
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 9
end index: 8
------------
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 10
```

```
end index: 9
------------
Quick Sort done: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Comparisons count: 64
Displacement count: 0
```

## Quick Sort B

```
Quick sorting: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
quick-sort: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
start index: 0
end index: 9
Sorting part: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
Pivot: 1
Small index: -1
Sorted part.
Part index: 0
Seperately quick sort parts
quick-sort: [1, 9, 8, 7, 6, 5, 4, 3, 2, 10]
start index: 0
end index: -1
------------
quick-sort: [1, 9, 8, 7, 6, 5, 4, 3, 2, 10]
start index: 1
end index: 9
Sorting part: [1, 9, 8, 7, 6, 5, 4, 3, 2, 10]
Pivot: 10
Small index: 0
Sorted part.
Part index: 9
Seperately quick sort parts
quick-sort: [1, 9, 8, 7, 6, 5, 4, 3, 2, 10]
start index: 1
end index: 8
Sorting part: [1, 9, 8, 7, 6, 5, 4, 3, 2, 10]
Pivot: 2
Small index: 0
Sorted part.
Part index: 1
Seperately quick sort parts
quick-sort: [1, 2, 8, 7, 6, 5, 4, 3, 9, 10]
start index: 1
end index: 0
------------
quick-sort: [1, 2, 8, 7, 6, 5, 4, 3, 9, 10]
start index: 2
end index: 8
Sorting part: [1, 2, 8, 7, 6, 5, 4, 3, 9, 10]
```

```
Pivot: 9
Small index: 1
Sorted part.
Part index: 8
Seperately quick sort parts
quick-sort: [1, 2, 8, 7, 6, 5, 4, 3, 9, 10]
start index: 2
end index: 7
Sorting part: [1, 2, 8, 7, 6, 5, 4, 3, 9, 10]
Pivot: 3
Small index: 1
Sorted part.
Part index: 2
Seperately quick sort parts
quick-sort: [1, 2, 3, 7, 6, 5, 4, 8, 9, 10]
start index: 2
end index: 1
------------
quick-sort: [1, 2, 3, 7, 6, 5, 4, 8, 9, 10]
start index: 3
end index: 7
Sorting part: [1, 2, 3, 7, 6, 5, 4, 8, 9, 10]
Pivot: 8
Small index: 2
Sorted part.
Part index: 7
Seperately quick sort parts
quick-sort: [1, 2, 3, 7, 6, 5, 4, 8, 9, 10]
start index: 3
end index: 6
Sorting part: [1, 2, 3, 7, 6, 5, 4, 8, 9, 10]
Pivot: 4
Small index: 2
Sorted part.
Part index: 3
Seperately quick sort parts
quick-sort: [1, 2, 3, 4, 6, 5, 7, 8, 9, 10]
start index: 3
end index: 2
------------
quick-sort: [1, 2, 3, 4, 6, 5, 7, 8, 9, 10]
start index: 4
end index: 6
Sorting part: [1, 2, 3, 4, 6, 5, 7, 8, 9, 10]
Pivot: 7
Small index: 3
Sorted part.
Part index: 6
```

```
Seperately quick sort parts
quick-sort: [1, 2, 3, 4, 6, 5, 7, 8, 9, 10]
start index: 4
end index: 5
Sorting part: [1, 2, 3, 4, 6, 5, 7, 8, 9, 10]
Pivot: 5
Small index: 3
Sorted part.
Part index: 4
Seperately quick sort parts
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 4
end index: 3
------------
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 5
end index: 5
------------
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 7
end index: 6
------------
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 8
end index: 7
------------
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 9
end index: 8
------------
quick-sort: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
start index: 10
end index: 9
------------
Quick Sort done: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Comparisons count: 64
Displacement count: 5
```

## Quick Sort C

```
Quick sorting: [5, 2, 13, 9, 1, 7, 6, 8, 1, 15, 4, 11]
quick-sort: [5, 2, 13, 9, 1, 7, 6, 8, 1, 15, 4, 11]
start index: 0
end index: 11
Sorting part: [5, 2, 13, 9, 1, 7, 6, 8, 1, 15, 4, 11]
Pivot: 11
Small index: -1
Sorted part.
```

```
Part index: 9
Seperately quick sort parts
quick-sort: [5, 2, 9, 1, 7, 6, 8, 1, 4, 11, 13, 15]
start index: 0
end index: 8
Sorting part: [5, 2, 9, 1, 7, 6, 8, 1, 4, 11, 13, 15]
Pivot: 4
Small index: -1
Sorted part.
Part index: 3
Seperately quick sort parts
quick-sort: [2, 1, 1, 4, 7, 6, 8, 9, 5, 11, 13, 15]
start index: 0
end index: 2
Sorting part: [2, 1, 1, 4, 7, 6, 8, 9, 5, 11, 13, 15]
Pivot: 1
Small index: -1
Sorted part.
Part index: 0
Seperately quick sort parts
quick-sort: [1, 1, 2, 4, 7, 6, 8, 9, 5, 11, 13, 15]
start index: 0
end index: -1
------------
quick-sort: [1, 1, 2, 4, 7, 6, 8, 9, 5, 11, 13, 15]
start index: 1
end index: 2
Sorting part: [1, 1, 2, 4, 7, 6, 8, 9, 5, 11, 13, 15]
Pivot: 2
Small index: 0
Sorted part.
Part index: 2
Seperately quick sort parts
quick-sort: [1, 1, 2, 4, 7, 6, 8, 9, 5, 11, 13, 15]
start index: 1
end index: 1
------------
quick-sort: [1, 1, 2, 4, 7, 6, 8, 9, 5, 11, 13, 15]
start index: 3
end index: 2
------------
quick-sort: [1, 1, 2, 4, 7, 6, 8, 9, 5, 11, 13, 15]
start index: 4
end index: 8
Sorting part: [1, 1, 2, 4, 7, 6, 8, 9, 5, 11, 13, 15]
Pivot: 5
Small index: 3
Sorted part.
```

```
Part index: 4
Seperately quick sort parts
quick-sort: [1, 1, 2, 4, 5, 6, 8, 9, 7, 11, 13, 15]
start index: 4
end index: 3
------------
quick-sort: [1, 1, 2, 4, 5, 6, 8, 9, 7, 11, 13, 15]
start index: 5
end index: 8
Sorting part: [1, 1, 2, 4, 5, 6, 8, 9, 7, 11, 13, 15]
Pivot: 7
Small index: 4
Sorted part.
Part index: 6
Seperately quick sort parts
quick-sort: [1, 1, 2, 4, 5, 6, 7, 9, 8, 11, 13, 15]
start index: 5
end index: 5
------------
quick-sort: [1, 1, 2, 4, 5, 6, 7, 9, 8, 11, 13, 15]
start index: 7
end index: 8
Sorting part: [1, 1, 2, 4, 5, 6, 7, 9, 8, 11, 13, 15]
Pivot: 8
Small index: 6
Sorted part.
Part index: 7
Seperately quick sort parts
quick-sort: [1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15]
start index: 7
end index: 6
------------
quick-sort: [1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15]
start index: 8
end index: 8
------------
quick-sort: [1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15]
start index: 10
end index: 11
Sorting part: [1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15]
Pivot: 15
Small index: 9
Sorted part.
Part index: 11
Seperately quick sort parts
quick-sort: [1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15]
start index: 10
end index: 10
```

```
------------
quick-sort: [1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15]
start index: 12
end index: 11
------------
Quick Sort done: [1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15]
Comparisons count: 48
Displacement count: 16
```

### Quick Sort D

```
Quick sorting: ['S', 'B', 'I', 'M', 'H', 'Q', 'C', 'L', 'R', 'E', 'P', 'K']
quick-sort: ['S', 'B', 'I', 'M', 'H', 'Q', 'C', 'L', 'R', 'E', 'P', 'K']
start index: 0
end index: 11
Sorting part: ['S', 'B', 'I', 'M', 'H', 'Q', 'C', 'L', 'R', 'E', 'P', 'K']
Pivot: K
Small index: -1
Sorted part.
Part index: 5
Seperately quick sort parts
quick-sort: ['B', 'I', 'H', 'C', 'E', 'K', 'M', 'L', 'R', 'S', 'P', 'Q']
start index: 0
end index: 4
Sorting part: ['B', 'I', 'H', 'C', 'E', 'K', 'M', 'L', 'R', 'S', 'P', 'Q']
Pivot: E
Small index: -1
Sorted part.
Part index: 2
Seperately quick sort parts
quick-sort: ['B', 'C', 'E', 'I', 'H', 'K', 'M', 'L', 'R', 'S', 'P', 'Q']
start index: 0
end index: 1
Sorting part: ['B', 'C', 'E', 'I', 'H', 'K', 'M', 'L', 'R', 'S', 'P', 'Q']
Pivot: C
Small index: -1
Sorted part.
Part index: 1
Seperately quick sort parts
quick-sort: ['B', 'C', 'E', 'I', 'H', 'K', 'M', 'L', 'R', 'S', 'P', 'Q']
start index: 0
end index: 0
------------
quick-sort: ['B', 'C', 'E', 'I', 'H', 'K', 'M', 'L', 'R', 'S', 'P', 'Q']
start index: 2
end index: 1
------------
quick-sort: ['B', 'C', 'E', 'I', 'H', 'K', 'M', 'L', 'R', 'S', 'P', 'Q']
```

```
start index: 3
end index: 4
Sorting part: ['B', 'C', 'E', 'I', 'H', 'K', 'M', 'L', 'R', 'S', 'P', 'Q']
Pivot: H
Small index: 2
Sorted part.
Part index: 3
Seperately quick sort parts
quick-sort: ['B', 'C', 'E', 'H', 'I', 'K', 'M', 'L', 'R', 'S', 'P', 'Q']
start index: 3
end index: 2
------------
quick-sort: ['B', 'C', 'E', 'H', 'I', 'K', 'M', 'L', 'R', 'S', 'P', 'Q']
start index: 4
end index: 4
------------
quick-sort: ['B', 'C', 'E', 'H', 'I', 'K', 'M', 'L', 'R', 'S', 'P', 'Q']
start index: 6
end index: 11
Sorting part: ['B', 'C', 'E', 'H', 'I', 'K', 'M', 'L', 'R', 'S', 'P', 'Q']
Pivot: Q
Small index: 5
Sorted part.
Part index: 9
Seperately quick sort parts
quick-sort: ['B', 'C', 'E', 'H', 'I', 'K', 'M', 'L', 'P', 'Q', 'R', 'S']
start index: 6
end index: 8
Sorting part: ['B', 'C', 'E', 'H', 'I', 'K', 'M', 'L', 'P', 'Q', 'R', 'S']
Pivot: P
Small index: 5
Sorted part.
Part index: 8
Seperately quick sort parts
quick-sort: ['B', 'C', 'E', 'H', 'I', 'K', 'M', 'L', 'P', 'Q', 'R', 'S']
start index: 6
end index: 7
Sorting part: ['B', 'C', 'E', 'H', 'I', 'K', 'M', 'L', 'P', 'Q', 'R', 'S']
Pivot: L
Small index: 5
Sorted part.
Part index: 6
Seperately quick sort parts
quick-sort: ['B', 'C', 'E', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
start index: 6
end index: 5
------------
quick-sort: ['B', 'C', 'E', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
```

```
start index: 7
end index: 7
------------
quick-sort: ['B', 'C', 'E', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
start index: 9
end index: 8
------------
quick-sort: ['B', 'C', 'E', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
start index: 10
end index: 11
Sorting part: ['B', 'C', 'E', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
Pivot: S
Small index: 9
Sorted part.
Part index: 11
Seperately quick sort parts
quick-sort: ['B', 'C', 'E', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
start index: 10
end index: 10
------------
quick-sort: ['B', 'C', 'E', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
start index: 12
end index: 11
------------
Quick Sort done: ['B', 'C', 'E', 'H', 'I', 'K', 'L', 'M', 'P', 'Q', 'R', 'S']
Comparisons count: 43
Displacement count: 12
```