Merge Sort

First we divide array to left and right sub arrays then sort them while merging

We ended the divide operation cause of nothing left to divide

Then compare leftSequence with rightSequence and fill outputSequence.

If an element put the outputSequence from lefft or right sequence, moves its cursor(index)

Q1:

*** A = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}**

*** B = { 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 }**

*** C = {5, 2, 13, 9, 1, 7, 6, 8, 1, 15, 4, 11}**

*** D = {‘S’, ‘B’, ‘I’, ‘M’, ‘H’, ‘Q’, ‘C’, ‘L’, ‘R’, ‘E’, ‘P’, ‘K’}**

A) = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

Merge Sort

First we divide array to left and right sub arrays then sort them while merging

We ended the divide operation cause of nothing left to divide

Then compare leftSequence with rightSequence and fill outputSequence.

If an element put the outputSequence from lefft or right sequence, moves its cursor(index)

Divided into two outputSequence : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,

leftSequence : 1, 2, 3, 4, 5,

rightSequence : 6, 7, 8, 9, 10,

Divided into two outputSequence : 1, 2, 3, 4, 5,

leftSequence : 1, 2,

rightSequence : 3, 4, 5,

```
 Divided into two outputSequence : 1, 2,
 leftSequence : 1,
 rightSequence : 2,

 Before Merge them,  outputSequence : 1, 2,
 leftSequence : 1,
 rightSequence : 2,

 compare index: 0 value: 1 at leftSequence and index: 0 value: 2 at rightSequence and fill index: 0 at outputSequence, value: 1 from leftSequence

 After comparisons leftSequence:


 After comparisons rightSequence:
 2,

 cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

 After merge outputSequence : 1, 2,

 Divided into two outputSequence : 3, 4, 5,
 leftSequence : 3,
 rightSequence : 4, 5,

 Divided into two outputSequence : 4, 5,
 leftSequence : 4,
 rightSequence : 5,
 Before Merge them,  outputSequence : 4, 5,
 leftSequence : 4,
 rightSequence : 5,

 compare index: 0 value: 4 at leftSequence and index: 0 value: 5 at rightSequence and fill index: 0 at outputSequence, value: 4 from leftSequence

 After comparisons leftSequence:


 After comparisons rightSequence:
 5,

 cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

 After merge outputSequence : 4, 5,

 Before Merge them,  outputSequence : 3, 4, 5,
 leftSequence : 3,
 rightSequence : 4, 5,

 compare index: 0 value: 3 at leftSequence and index: 0 value: 4 at rightSequence and fill index: 0 at outputSequence, value: 3 from leftSequence

 After comparisons leftSequence:


 After comparisons rightSequence:
```

4, 5,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : 3, 4, 5,

Before Merge them, outputSequence : 1, 2, 3, 4, 5,

leftSequence : 1, 2,

rightSequence : 3, 4, 5,

compare index: 0 value: 1 at leftSequence and index: 0 value: 3 at rightSequence and fill index: 0 at outputSequence, value: 1 from leftSequence

compare index: 1 value: 2 at leftSequence and index: 0 value: 3 at rightSequence and fill index: 1 at outputSequence, value: 2 from leftSequence

After comparisons leftSequence:

After comparisons rightSequence:

3, 4, 5,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : 1, 2, 3, 4, 5,

Divided into two outputSequence : 6, 7, 8, 9, 10,

leftSequence : 6, 7,

rightSequence : 8, 9, 10,

Divided into two outputSequence : 6, 7,

leftSequence : 6,

rightSequence : 7,

Before Merge them, outputSequence : 6, 7,

leftSequence : 6,

rightSequence : 7,

compare index: 0 value: 6 at leftSequence and index: 0 value: 7 at rightSequence and fill index: 0 at outputSequence, value: 6 from leftSequence

After comparisons leftSequence:

After comparisons rightSequence:

7,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : 6, 7,

Divided into two outputSequence : 8, 9, 10,

leftSequence : 8,

rightSequence : 9, 10,

Divided into two outputSequence : 9, 10,

leftSequence : 9,

rightSequence : 10,

Before Merge them, outputSequence : 9, 10,

leftSequence : 9,

rightSequence : 10,

compare index: 0 value: 9 at leftSequence and index: 0 value: 10 at rightSequence and fill index: 0 at outputSequence, value: 9 from leftSequence

After comparisons leftSequence:

After comparisons rightSequence:

10,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : 9, 10,

Before Merge them, outputSequence : 8, 9, 10,

leftSequence : 8,

rightSequence : 9, 10,

compare index: 0 value: 8 at leftSequence and index: 0 value: 9 at rightSequence and fill index: 0 at outputSequence, value: 8 from leftSequence

After comparisons leftSequence:

After comparisons rightSequence:

9, 10,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : 8, 9, 10,

Before Merge them, outputSequence : 6, 7, 8, 9, 10,

leftSequence : 6, 7,

rightSequence : 8, 9, 10,

compare index: 0 value: 6 at leftSequence and index: 0 value: 8 at rightSequence and fill index: 0 at outputSequence, value: 6 from leftSequence

compare index: 1 value: 7 at leftSequence and index: 0 value: 8 at rightSequence and fill index: 1 at outputSequence, value: 7 from leftSequence

After comparisons leftSequence:

After comparisons rightSequence:

8, 9, 10,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : 6, 7, 8, 9, 10,

Before Merge them, outputSequence : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,

leftSequence : 1, 2, 3, 4, 5,

rightSequence : 6, 7, 8, 9, 10,

compare index: 0 value: 1 at leftSequence and index: 0 value: 6 at rightSequence and fill index: 0 at outputSequence, value: 1 from leftSequence

compare index: 1 value: 2 at leftSequence and index: 0 value: 6 at rightSequence and fill index: 1 at outputSequence, value: 2 from leftSequence

compare index: 2 value: 3 at leftSequence and index: 0 value: 6 at rightSequence and fill index: 2 at outputSequence, value: 3 from leftSequence

compare index: 3 value: 4 at leftSequence and index: 0 value: 6 at rightSequence and fill index: 3 at outputSequence, value: 4 from leftSequence

compare index: 4 value: 5 at leftSequence and index: 0 value: 6 at rightSequence and fill index: 4 at outputSequence, value: 5 from leftSequence

After comparisons leftSequence:

After comparisons rightSequence:

6, 7, 8, 9, 10,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,

B) = { 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 }

Merge Sort

First we divide array to left and right sub arrays then sort them while merging

We ended the divide operation cause of nothing left to divide

Then compare leftSequence with rightSequence and fill outputSequence.

If an element put the outputSequence from lefft or right sequence, moves its cursor(index)

Divided into two outputSequence : 10, 9, 8, 7, 6, 5, 4, 3, 2, 1,

leftSequence : 10, 9, 8, 7, 6,

rightSequence : 5, 4, 3, 2, 1,

Divided into two outputSequence : 10, 9, 8, 7, 6,

leftSequence : 10, 9,

rightSequence : 8, 7, 6,

Divided into two outputSequence : 10, 9,

leftSequence : 10,

rightSequence : 9,

Before Merge them, outputSequence : 10, 9,

leftSequence : 10,

rightSequence : 9,

compare index: 0 value: 10 at leftSequence and index: 0 value: 9 at rightSequence and fill index: 0 at outputSequence, value: 9 from rightSequence

After comparisons leftSequence:

10,

After comparisons rightSequence:

```
 cause of there is no element at rightSequence, fill other elements at the leftSequence to
outputSequence in order

 After merge outputSequence : 9, 10,

 Divided into two outputSequence : 8, 7, 6,
 leftSequence : 8,
 rightSequence : 7, 6,

 Divided into two outputSequence : 7, 6,
 leftSequence : 7,
 rightSequence : 6,

 Before Merge them,  outputSequence : 7, 6,
 leftSequence : 7,
 rightSequence : 6,

 compare index: 0 value: 7 at leftSequence and index: 0 value: 6 at rightSequence and fill index: 0 at
outputSequence, value: 6 from rightSequence

 After comparisons leftSequence:
 7,

 After comparisons rightSequence:

 cause of there is no element at rightSequence, fill other elements at the leftSequence to
outputSequence in order

 After merge outputSequence : 6, 7,

 Before Merge them,  outputSequence : 8, 7, 6,
 leftSequence : 8,
```

rightSequence : 6, 7,

compare index: 0 value: 8 at leftSequence and index: 0 value: 6 at rightSequence and fill index: 0 at outputSequence, value: 6 from rightSequence

compare index: 0 value: 8 at leftSequence and index: 1 value: 7 at rightSequence and fill index: 1 at outputSequence, value: 7 from rightSequence

After comparisons leftSequence:

8,

After comparisons rightSequence:

cause of there is no element at rightSequence, fill other elements at the leftSequence to outputSequence in order

After merge outputSequence : 6, 7, 8,

Before Merge them, outputSequence : 10, 9, 8, 7, 6,

leftSequence : 9, 10,

rightSequence : 6, 7, 8,

compare index: 0 value: 9 at leftSequence and index: 0 value: 6 at rightSequence and fill index: 0 at outputSequence, value: 6 from rightSequence

compare index: 0 value: 9 at leftSequence and index: 1 value: 7 at rightSequence and fill index: 1 at outputSequence, value: 7 from rightSequence

compare index: 0 value: 9 at leftSequence and index: 2 value: 8 at rightSequence and fill index: 2 at outputSequence, value: 8 from rightSequence

After comparisons leftSequence:

9, 10,

After comparisons rightSequence:

cause of there is no element at rightSequence, fill other elements at the leftSequence to outputSequence in order

After merge outputSequence : 6, 7, 8, 9, 10,

Divided into two outputSequence : 5, 4, 3, 2, 1,

leftSequence : 5, 4,

rightSequence : 3, 2, 1,

Divided into two outputSequence : 5, 4,

leftSequence : 5,

rightSequence : 4,

Before Merge them, outputSequence : 5, 4,

leftSequence : 5,

rightSequence : 4,

compare index: 0 value: 5 at leftSequence and index: 0 value: 4 at rightSequence and fill index: 0 at outputSequence, value: 4 from rightSequence

After comparisons leftSequence:

5,

After comparisons rightSequence:

cause of there is no element at rightSequence, fill other elements at the leftSequence to outputSequence in order

After merge outputSequence : 4, 5,

Divided into two outputSequence : 3, 2, 1,

leftSequence : 3,

rightSequence : 2, 1,

Divided into two outputSequence : 2, 1,

leftSequence : 2,

rightSequence : 1,

Before Merge them, outputSequence : 2, 1,

leftSequence : 2,

rightSequence : 1,

compare index: 0 value: 2 at leftSequence and index: 0 value: 1 at rightSequence and fill index: 0 at outputSequence, value: 1 from rightSequence

After comparisons leftSequence:

2,

After comparisons rightSequence:

cause of there is no element at rightSequence, fill other elements at the leftSequence to outputSequence in order

After merge outputSequence : 1, 2,

Before Merge them, outputSequence : 3, 2, 1,

leftSequence : 3,

rightSequence : 1, 2,

compare index: 0 value: 3 at leftSequence and index: 0 value: 1 at rightSequence and fill index: 0 at outputSequence, value: 1 from rightSequence

compare index: 0 value: 3 at leftSequence and index: 1 value: 2 at rightSequence and fill index: 1 at outputSequence, value: 2 from rightSequence

```
 After comparisons leftSequence:

 3,


 After comparisons rightSequence:


 cause of there is no element at rightSequence, fill other elements at the leftSequence to outputSequence in order


 After merge outputSequence : 1, 2, 3,


 Before Merge them,  outputSequence : 5, 4, 3, 2, 1,

 leftSequence : 4, 5,

 rightSequence : 1, 2, 3,


 compare index: 0 value: 4 at leftSequence and index: 0 value: 1 at rightSequence and fill index: 0 at outputSequence, value: 1 from rightSequence

 compare index: 0 value: 4 at leftSequence and index: 1 value: 2 at rightSequence and fill index: 1 at outputSequence, value: 2 from rightSequence

 compare index: 0 value: 4 at leftSequence and index: 2 value: 3 at rightSequence and fill index: 2 at outputSequence, value: 3 from rightSequence


 After comparisons leftSequence:

 4,  5,


 After comparisons rightSequence:


 cause of there is no element at rightSequence, fill other elements at the leftSequence to outputSequence in order


 After merge outputSequence : 1, 2, 3, 4, 5,


 Before Merge them,  outputSequence : 10, 9, 8, 7, 6, 5, 4, 3, 2, 1,

 leftSequence : 6, 7, 8, 9, 10,
```

rightSequence : 1, 2, 3, 4, 5,

compare index: 0 value: 6 at leftSequence and index: 0 value: 1 at rightSequence and fill index: 0 at outputSequence, value: 1 from rightSequence

compare index: 0 value: 6 at leftSequence and index: 1 value: 2 at rightSequence and fill index: 1 at outputSequence, value: 2 from rightSequence

compare index: 0 value: 6 at leftSequence and index: 2 value: 3 at rightSequence and fill index: 2 at outputSequence, value: 3 from rightSequence

compare index: 0 value: 6 at leftSequence and index: 3 value: 4 at rightSequence and fill index: 3 at outputSequence, value: 4 from rightSequence

compare index: 0 value: 6 at leftSequence and index: 4 value: 5 at rightSequence and fill index: 4 at outputSequence, value: 5 from rightSequence

After comparisons leftSequence:

6, 7, 8, 9, 10,

After comparisons rightSequence:

cause of there is no element at rightSequence, fill other elements at the leftSequence to outputSequence in order

After merge outputSequence : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,

C) = {5, 2, 13, 9, 1, 7, 6, 8, 1, 15, 4, 11}

Merge Sort

Divided into two outputSequence : 5, 2, 13, 9, 1, 7, 6, 8, 1, 15, 4, 11,

leftSequence : 5, 2, 13, 9, 1, 7,

rightSequence : 6, 8, 1, 15, 4, 11,

Divided into two outputSequence : 5, 2, 13, 9, 1, 7,

leftSequence : 5, 2, 13,

rightSequence : 9, 1, 7,

Divided into two outputSequence : 5, 2, 13,

leftSequence : 5,

rightSequence : 2, 13,

Divided into two outputSequence : 2, 13,

leftSequence : 2,

rightSequence : 13,

Before Merge them, outputSequence : 2, 13,

leftSequence : 2,

rightSequence : 13,

compare index: 0 value: 2 at leftSequence and index: 0 value: 13 at rightSequence and fill index: 0 at outputSequence, value: 2 from leftSequence

After comparisons leftSequence:

After comparisons rightSequence:

13,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : 2, 13,

Before Merge them, outputSequence : 5, 2, 13,

leftSequence : 5,

rightSequence : 2, 13,

compare index: 0 value: 5 at leftSequence and index: 0 value: 2 at rightSequence and fill index: 0 at outputSequence, value: 2 from rightSequence compare index: 0 value: 5 at leftSequence and index: 1 value: 13 at rightSequence and fill index: 1 at outputSequence, value: 5 from leftSequence

After comparisons leftSequence:

After comparisons rightSequence:

13,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : 2, 5, 13,

Divided into two outputSequence : 9, 1, 7,

leftSequence : 9,

rightSequence : 1, 7,

Divided into two outputSequence : 1, 7,

leftSequence : 1,

rightSequence : 7,

Before Merge them, outputSequence : 1, 7,

leftSequence : 1,

rightSequence : 7,

compare index: 0 value: 1 at leftSequence and index: 0 value: 7 at rightSequence and fill index: 0 at outputSequence, value: 1 from leftSequence

After comparisons leftSequence:

```
 After comparisons rightSequence:
 7,

 cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

 After merge outputSequence : 1, 7,

 Before Merge them,  outputSequence : 9, 1, 7,
 leftSequence : 9,
 rightSequence : 1, 7,

 compare index: 0 value: 9 at leftSequence and index: 0 value: 1 at rightSequence and fill index: 0 at outputSequence, value: 1 from rightSequence
 compare index: 0 value: 9 at leftSequence and index: 1 value: 7 at rightSequence and fill index: 1 at outputSequence, value: 7 from rightSequence

 After comparisons leftSequence:
 9,

 After comparisons rightSequence:

 cause of there is no element at rightSequence, fill other elements at the leftSequence to outputSequence in order

 After merge outputSequence : 1, 7, 9,

 Before Merge them,  outputSequence : 5, 2, 13, 9, 1, 7,
 leftSequence : 2, 5, 13,
 rightSequence : 1, 7, 9,

 compare index: 0 value: 2 at leftSequence and index: 0 value: 1 at rightSequence and fill index: 0 at outputSequence, value: 1 from rightSequence
 compare index: 0 value: 2 at leftSequence and index: 1 value: 7 at rightSequence and fill index: 1 at
outputSequence, value: 2 from leftSequence

 compare index: 1 value: 5 at leftSequence and index: 1 value: 7 at rightSequence and fill index: 2 at
outputSequence, value: 5 from leftSequence

 compare index: 2 value: 13 at leftSequence and index: 1 value: 7 at rightSequence and fill index: 3 at
outputSequence, value: 7 from rightSequence

 compare index: 2 value: 13 at leftSequence and index: 2 value: 9 at rightSequence and fill index: 4 at
outputSequence, value: 9 from rightSequence


 After comparisons leftSequence:

 13,


 After comparisons rightSequence:


 cause of there is no element at rightSequence, fill other elements at the leftSequence to
outputSequence in order


 After merge outputSequence : 1, 2, 5, 7, 9, 13,


 Divided into two outputSequence : 6, 8, 1, 15, 4, 11,

 leftSequence : 6, 8, 1,

 rightSequence : 15, 4, 11,


 Divided into two outputSequence : 6, 8, 1,

 leftSequence : 6,

 rightSequence : 8, 1,


 Divided into two outputSequence : 8, 1,

 leftSequence : 8,

 rightSequence : 1,


 Before Merge them,  outputSequence : 8, 1,

 leftSequence : 8,
```

rightSequence : 1,

compare index: 0 value: 8 at leftSequence and index: 0 value: 1 at rightSequence and fill index: 0 at outputSequence, value: 1 from rightSequence

After comparisons leftSequence:

8,

After comparisons rightSequence:

cause of there is no element at rightSequence, fill other elements at the leftSequence to outputSequence in order

After merge outputSequence : 1, 8,

Before Merge them, outputSequence : 6, 8, 1,

leftSequence : 6,

rightSequence : 1, 8,

compare index: 0 value: 6 at leftSequence and index: 0 value: 1 at rightSequence and fill index: 0 at outputSequence, value: 1 from rightSequence

compare index: 0 value: 6 at leftSequence and index: 1 value: 8 at rightSequence and fill index: 1 at outputSequence, value: 6 from leftSequence

After comparisons leftSequence:

After comparisons rightSequence:

8,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

```
 After merge outputSequence : 1, 6, 8,


 Divided into two outputSequence : 15, 4, 11,

 leftSequence : 15,

 rightSequence : 4, 11,


 Divided into two outputSequence : 4, 11,

 leftSequence : 4,

 rightSequence : 11,


 Before Merge them,  outputSequence : 4, 11,

 leftSequence : 4,

 rightSequence : 11,


 compare index: 0 value: 4 at leftSequence and index: 0 value: 11 at rightSequence and fill index: 0 at outputSequence, value: 4 from leftSequence


 After comparisons leftSequence:



 After comparisons rightSequence:

 11,


 cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order


 After merge outputSequence : 4, 11,


 Before Merge them,  outputSequence : 15, 4, 11,

 leftSequence : 15,

 rightSequence : 4, 11,
 compare index: 0 value: 15 at leftSequence and index: 0 value: 4 at rightSequence and fill index: 0 at
outputSequence, value: 4 from rightSequence

 compare index: 0 value: 15 at leftSequence and index: 1 value: 11 at rightSequence and fill index: 1
at outputSequence, value: 11 from rightSequence


 After comparisons leftSequence:

 15,


 After comparisons rightSequence:


 cause of there is no element at rightSequence, fill other elements at the leftSequence to
outputSequence in order


 After merge outputSequence : 4, 11, 15,


 Before Merge them,  outputSequence : 6, 8, 1, 15, 4, 11,

 leftSequence : 1, 6, 8,

 rightSequence : 4, 11, 15,


 compare index: 0 value: 1 at leftSequence and index: 0 value: 4 at rightSequence and fill index: 0 at
outputSequence, value: 1 from leftSequence

 compare index: 1 value: 6 at leftSequence and index: 0 value: 4 at rightSequence and fill index: 1 at
outputSequence, value: 4 from rightSequence

 compare index: 1 value: 6 at leftSequence and index: 1 value: 11 at rightSequence and fill index: 2 at
outputSequence, value: 6 from leftSequence

 compare index: 2 value: 8 at leftSequence and index: 1 value: 11 at rightSequence and fill index: 3 at
outputSequence, value: 8 from leftSequence


 After comparisons leftSequence:



 After comparisons rightSequence:

 11,  15,
```

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : 1, 4, 6, 8, 11, 15,

Before Merge them, outputSequence : 5, 2, 13, 9, 1, 7, 6, 8, 1, 15, 4, 11,

leftSequence : 1, 2, 5, 7, 9, 13,

rightSequence : 1, 4, 6, 8, 11, 15,

compare index: 0 value: 1 at leftSequence and index: 0 value: 1 at rightSequence and fill index: 0 at outputSequence, value: 1 from rightSequence

compare index: 0 value: 1 at leftSequence and index: 1 value: 4 at rightSequence and fill index: 1 at outputSequence, value: 1 from leftSequence

compare index: 1 value: 2 at leftSequence and index: 1 value: 4 at rightSequence and fill index: 2 at outputSequence, value: 2 from leftSequence

compare index: 2 value: 5 at leftSequence and index: 1 value: 4 at rightSequence and fill index: 3 at outputSequence, value: 4 from rightSequence

compare index: 2 value: 5 at leftSequence and index: 2 value: 6 at rightSequence and fill index: 4 at outputSequence, value: 5 from leftSequence

compare index: 3 value: 7 at leftSequence and index: 2 value: 6 at rightSequence and fill index: 5 at outputSequence, value: 6 from rightSequence

compare index: 3 value: 7 at leftSequence and index: 3 value: 8 at rightSequence and fill index: 6 at outputSequence, value: 7 from leftSequence

compare index: 4 value: 9 at leftSequence and index: 3 value: 8 at rightSequence and fill index: 7 at outputSequence, value: 8 from rightSequence

compare index: 4 value: 9 at leftSequence and index: 4 value: 11 at rightSequence and fill index: 8 at outputSequence, value: 9 from leftSequence

compare index: 5 value: 13 at leftSequence and index: 4 value: 11 at rightSequence and fill index: 9 at outputSequence, value: 11 from rightSequence

compare index: 5 value: 13 at leftSequence and index: 5 value: 15 at rightSequence and fill index: 10 at outputSequence, value: 13 from leftSequence

After comparisons leftSequence:

After comparisons rightSequence:

15,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : 1, 1, 2, 4, 5, 6, 7, 8, 9, 11, 13, 15,

D) = {‘S’, ‘B’, ‘I’, ‘M’, ‘H’, ‘Q’, ‘C’, ‘L’, ‘R’, ‘E’, ‘P’, ‘K’}

Merge Sort

Divided into two outputSequence : S, B, I, M, H, Q, C, L, R, E, P, K,

leftSequence : S, B, I, M, H, Q,

rightSequence : C, L, R, E, P, K,

Divided into two outputSequence : S, B, I, M, H, Q,

leftSequence : S, B, I,

rightSequence : M, H, Q,

Divided into two outputSequence : S, B, I,

leftSequence : S,

rightSequence : B, I,

Divided into two outputSequence : B, I,

leftSequence : B,

rightSequence : I,

Before Merge them, outputSequence : B, I,

leftSequence : B,

rightSequence : I,

compare index: 0 value: B at leftSequence and index: 0 value: I at rightSequence and fill index: 0 at outputSequence, value: B from leftSequence

After comparisons leftSequence:

After comparisons rightSequence:

I,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : B, I,

Before Merge them, outputSequence : S, B, I,

leftSequence : S,

rightSequence : B, I,

compare index: 0 value: S at leftSequence and index: 0 value: B at rightSequence and fill index: 0 at outputSequence, value: B from rightSequence

compare index: 0 value: S at leftSequence and index: 1 value: I at rightSequence and fill index: 1 at outputSequence, value: I from rightSequence

After comparisons leftSequence:

S,

After comparisons rightSequence:

cause of there is no element at rightSequence, fill other elements at the leftSequence to outputSequence in order

```
After merge outputSequence : B, I, S,


Divided into two outputSequence : M, H, Q,

leftSequence : M,

rightSequence : H, Q,


Divided into two outputSequence : H, Q,

leftSequence : H,

rightSequence : Q,


Before Merge them,  outputSequence : H, Q,

leftSequence : H,

rightSequence : Q,


compare index: 0 value: H at leftSequence and index: 0 value: Q at rightSequence and fill index: 0 at outputSequence, value: H from leftSequence


After comparisons leftSequence:



After comparisons rightSequence:

Q,


cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order


After merge outputSequence : H, Q,


Before Merge them,  outputSequence : M, H, Q,

leftSequence : M,
```

rightSequence : H, Q,

compare index: 0 value: M at leftSequence and index: 0 value: H at rightSequence and fill index: 0 at outputSequence, value: H from rightSequence

compare index: 0 value: M at leftSequence and index: 1 value: Q at rightSequence and fill index: 1 at outputSequence, value: M from leftSequence

After comparisons leftSequence:

After comparisons rightSequence:

Q,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : H, M, Q,

Before Merge them, outputSequence : S, B, I, M, H, Q,

leftSequence : B, I, S,

rightSequence : H, M, Q,

compare index: 0 value: B at leftSequence and index: 0 value: H at rightSequence and fill index: 0 at outputSequence, value: B from leftSequence

compare index: 1 value: I at leftSequence and index: 0 value: H at rightSequence and fill index: 1 at outputSequence, value: H from rightSequence

compare index: 1 value: I at leftSequence and index: 1 value: M at rightSequence and fill index: 2 at outputSequence, value: I from leftSequence

compare index: 2 value: S at leftSequence and index: 1 value: M at rightSequence and fill index: 3 at outputSequence, value: M from rightSequence

compare index: 2 value: S at leftSequence and index: 2 value: Q at rightSequence and fill index: 4 at outputSequence, value: Q from rightSequence

After comparisons leftSequence:

S,

After comparisons rightSequence:

cause of there is no element at rightSequence, fill other elements at the leftSequence to outputSequence in order

After merge outputSequence : B, H, I, M, Q, S,

Divided into two outputSequence : C, L, R, E, P, K,

leftSequence : C, L, R,

rightSequence : E, P, K,

Divided into two outputSequence : C, L, R,

leftSequence : C,

rightSequence : L, R,

Divided into two outputSequence : L, R,

leftSequence : L,

rightSequence : R,

Before Merge them, outputSequence : L, R,

leftSequence : L,

rightSequence : R,

compare index: 0 value: L at leftSequence and index: 0 value: R at rightSequence and fill index: 0 at outputSequence, value: L from leftSequence

After comparisons leftSequence:

After comparisons rightSequence:

R,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : L, R,

Before Merge them, outputSequence : C, L, R,

leftSequence : C,

rightSequence : L, R,

compare index: 0 value: C at leftSequence and index: 0 value: L at rightSequence and fill index: 0 at outputSequence, value: C from leftSequence

After comparisons leftSequence:

After comparisons rightSequence:

L, R,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : C, L, R,

Divided into two outputSequence : E, P, K, leftSequence : E,

rightSequence : P, K,

Divided into two outputSequence : P, K,

leftSequence : P,

rightSequence : K,

Before Merge them, outputSequence : P, K,

leftSequence : P,

rightSequence : K,

compare index: 0 value: P at leftSequence and index: 0 value: K at rightSequence and fill index: 0 at outputSequence, value: K from rightSequence

After comparisons leftSequence:

P,

After comparisons rightSequence:

cause of there is no element at rightSequence, fill other elements at the leftSequence to outputSequence in order

After merge outputSequence : K, P,

Before Merge them, outputSequence : E, P, K,

leftSequence : E,

rightSequence : K, P,

compare index: 0 value: E at leftSequence and index: 0 value: K at rightSequence and fill index: 0 at outputSequence, value: E from leftSequence

After comparisons leftSequence:

After comparisons rightSequence:

K, P,

cause of there is no element at leftSequence, fill other elements at the rightSequence to outputSequence in order

After merge outputSequence : E, K, P,

Before Merge them, outputSequence : C, L, R, E, P, K,

leftSequence : C, L, R,

rightSequence : E, K, P,

compare index: 0 value: C at leftSequence and index: 0 value: E at rightSequence and fill index: 0 at outputSequence, value: C from leftSequence

compare index: 1 value: L at leftSequence and index: 0 value: E at rightSequence and fill index: 1 at outputSequence, value: E from rightSequence

compare index: 1 value: L at leftSequence and index: 1 value: K at rightSequence and fill index: 2 at outputSequence, value: K from rightSequence

compare index: 1 value: L at leftSequence and index: 2 value: P at rightSequence and fill index: 3 at outputSequence, value: L from leftSequence

compare index: 2 value: R at leftSequence and index: 2 value: P at rightSequence and fill index: 4 at outputSequence, value: P from rightSequence

After comparisons leftSequence:

R,

After comparisons rightSequence:

cause of there is no element at rightSequence, fill other elements at the leftSequence to outputSequence in order

After merge outputSequence : C, E, K, L, P, R,

Before Merge them, outputSequence : S, B, I, M, H, Q, C, L, R, E, P, K,

leftSequence : B, H, I, M, Q, S,

rightSequence : C, E, K, L, P, R,

compare index: 0 value: B at leftSequence and index: 0 value: C at rightSequence and fill index: 0 at outputSequence, value: B from leftSequence

compare index: 1 value: H at leftSequence and index: 0 value: C at rightSequence and fill index: 1 at outputSequence, value: C from rightSequence

compare index: 1 value: H at leftSequence and index: 1 value: E at rightSequence and fill index: 2 at outputSequence, value: E from rightSequence

compare index: 1 value: H at leftSequence and index: 2 value: K at rightSequence and fill index: 3 at outputSequence, value: H from leftSequence

compare index: 2 value: I at leftSequence and index: 2 value: K at rightSequence and fill index: 4 at outputSequence, value: I from leftSequence

compare index: 3 value: M at leftSequence and index: 2 value: K at rightSequence and fill index: 5 at outputSequence, value: K from rightSequence

compare index: 3 value: M at leftSequence and index: 3 value: L at rightSequence and fill index: 6 at outputSequence, value: L from rightSequence

compare index: 3 value: M at leftSequence and index: 4 value: P at rightSequence and fill index: 7 at outputSequence, value: M from leftSequence

compare index: 4 value: Q at leftSequence and index: 4 value: P at rightSequence and fill index: 8 at outputSequence, value: P from rightSequence

compare index: 4 value: Q at leftSequence and index: 5 value: R at rightSequence and fill index: 9 at outputSequence, value: Q from leftSequence

compare index: 5 value: S at leftSequence and index: 5 value: R at rightSequence and fill index: 10 at outputSequence, value: R from rightSequence

After comparisons leftSequence:

S,

After comparisons rightSequence:

cause of there is no element at rightSequence, fill other elements at the leftSequence to outputSequence in order

After merge outputSequence : B, C, E, H, I, K, L, M, P, Q, R, S,