Git Hub Link:

Merge Sort:

Initial List:

4

89

23

118

15

67

7

29

6

51

2

30

1

11

First Divide:

7

67

15

4

89

23

118

29

6

51

2

30

1

11

Second Divide

4

89

23

118

15

67

7

2

51

6

29

30

11

1

Third Divide:

89

23

4

118

15

67

7

29

6

51

2

30

11

1

Fourth Divide:

118

23

89

4

11

1

30

2

51

6

29

15

67

7

Merge

89

23

118

4

15

67

7

29

6

51

2

11

1

30

Merge

118

89

23

4

51

67

15

7

29

6

2

30

11

1

Merge

118

89

67

23

15

7

4

51

30

29

6

11

2

1

Merge

118

89

67

51

30

29

23

15

11

7

6

4

2

1

The goal here is to showcase each step of the Merge Sort Algorithm. The principal for merge sort is divide and conquer, so the first steps are dividing up the numbers evenly, then the proceeding steps merge the numbers and sort between smaller groups of values in an array.

Quick Sort:

7

67

15

29

6

51

2

30

1

11

4

89

23

118

Pivot

11

4

7

6

2

1

23

89

118

15

67

29

30

51

Left sub–List First number as pivot: Right sub-List First number as pivot:

30

118

89

67

51

15

23

29

4

7

4

6

2

4

7

6

2

1

29

51

118

89

67

15

23

Sorted

67

89

118

6

7

Result:

89

118

7

6

4

2

1

Result

118

89

67

51

29

23

15

Total Result after combining sub lists:

7

89

118

67

51

23

15

29

11

6

4

2

1

For the Quick Sort Function, the goal was again to divide and conquer, but for this method you continuously compare all the numbers in a list a subsequent sub lists to one pivot number to see if the values were greater or less then the pivot. As the sub lists continue you are eventually left with a list of ordered numbers, and the previous pivots are added back into a sorted list depending on the numbers in the sub list being the list of lower numbers or higher numbers compared to the pivot.