

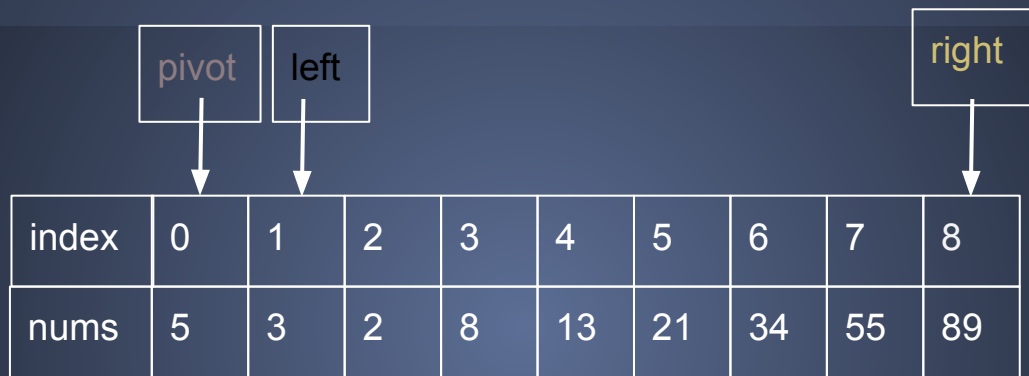
2.0 merge sort general ideas

```
int mid = ( lo + hi ) / 2;  
mergeSort(data, lo, mid);  
mergeSort(data, mid + 1, hi);  
merge(data, lo, mid, mid + 1, hi, tmp)  
copy sorted num from tmp to data
```

Merge: scan two arrays from head -> tail

<https://github.com/xxu46/codingAbility/blob/master/ThirdClass.java>
330 lines

3.0 quick sort, 不断地固定下位置



The diagram illustrates the partitioning step of the quick sort algorithm. It features a table with two rows: 'index' and 'nums'. Above the table, three boxes labeled 'pivot', 'left', and 'right' have arrows pointing to specific indices. The 'pivot' box points to index 0, the 'left' box points to index 1, and the 'right' box points to index 8. The table contains the following data:

| | | | | | | | | | |
|-------|---|---|---|---|----|----|----|----|----|
| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| nums | 5 | 3 | 2 | 8 | 13 | 21 | 34 | 55 | 89 |

1. left, right are two borders, records numbers that are less than and greater than pivot value. finally swap pivot and left. we find pivot relative position has already 固定下来。接下来是 recursively use quick sort pivot

| | | | | | | | |
|----|----|----|----|----|----|----|----|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 15 | 12 | 13 | 11 | 20 | 15 | 22 | 14 |



SPLIT

| | | | | | | | |
|----|----|----|----|----|----|----|----|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 13 | 12 | 14 | 11 | 15 | 20 | 15 | 22 |

DIVIDE
AND
CONQUER



Recursive
Sort

Recursive
Sort



| | | | | | | | |
|----|----|----|----|----|----|----|----|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 11 | 12 | 13 | 14 | 15 | 15 | 20 | 22 |