Divide And Conquer

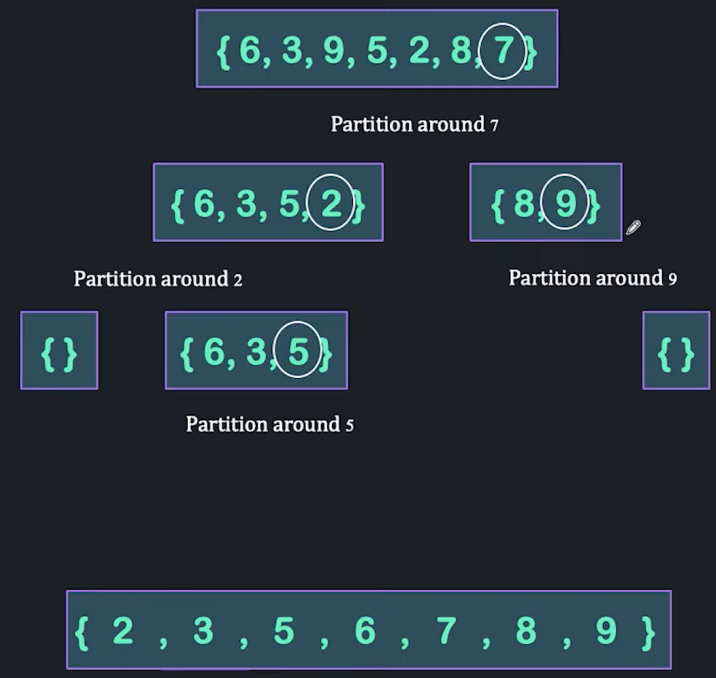
Quick sort

Idea: Select a random pivot, put it in its correct position, and sort the left and right part recursively.

Time Complexity: Avg = O(N logN), Worst Case = $O(N^2)$

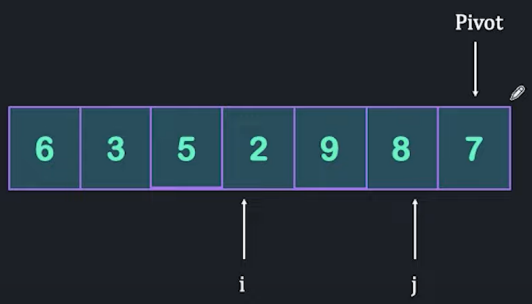
```
void swap(int arr[], int i, int j) {
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;
}

int partition(int arr[], int l, int r) {

    int pivot = arr[r];
    int i = l - 1;

    for (int j = l; j < r; j++) {
        if (arr[j] < pivot) {
            i++;
            swap(arr, i, j);
        }
    }
    swap(arr, i + 1, r);
    return i + 1;
}

void quickSort(int arr[], int l, int r) {

    if (l < r) {

        int pi = partition(arr, l, r);
        quickSort(arr, l, pi - 1);
        quickSort(arr, pi + 1, r);

    }
}
```

```
QuickSort(arr[], l, r){

  if( l < r ){

    int pi = partition(arr[], l, r)

    QuickSort(arr[], l, pi-1)
    QuickSort(arr[], pi+1, r)

  }

}
```



Quick Sort | Code and Explanation | C++ Course - 19.2

```
Partition(arr[], l, r){

  pivot = arr[r];
  i = l - 1
  for j=l to r-1
      if arr[j] < pivot
          i++;
          swap(i,j)

  swap(i+1, r)
  return i+1

}
```



7:53 / 15:23

Quick Sort Complexity:

Depends on pivot.

1. In best case, pivot would be median element.
2. In worst case, pivot could be end element.