

```
public static void quickSort(int arr[], int low, int high) {  
    if (low < high) {  
        int pi = partition(arr, low, high);  
  
        quickSort(arr, low, pi - 1);  
        quickSort(arr, pi + 1, high);  
    }  
}
```

## Time Complexity

Worst :  $O(n^2)$

Average :  $O(n \log n)$

```
public static int partition(int[] arr, int low, int high) {  
    int pivot = arr[high];  
    int i = low-1;  
  
    for(int j=low; j<high; j++) {  
        if (arr[j] < pivot) {  
            i++;  
            //swap  
            int temp = arr[i];  
            arr[i] = arr[j];  
            arr[j] = temp;  
        }  
    }  
    //swap with pivot  
    i++;  
    int temp = arr[i];  
    arr[i] = arr[high];  
    arr[high] = temp;  
    return i;  
}
```

### Important

**Worst case occurs  
when pivot is always  
the smallest or the  
largest element.**