- **全局排序**，O(n*lg(n))
- **局部排序**，只排序 TopK 个数，O(n*k)
- **堆**，TopK 个数也不排序了，O(n*lg(k))
- ~~分治法，每个分支“都要”递归，例如：快速排序，O(n*lg(n))~~
- ~~减治法，“只要”递归一个分支，例如：三分查找 O(lg(n))，随机选择 O(n)~~
- **随机选择**+partition
- **比特位图计数**

随机选择并排序/基于快速排序：

```
#include<iostream>
using namespace std;

int findK(int nums[],int k,int start,int end) {
    int low=start;
    int high=end;
    int temp=nums[low];//枢纽点
    while(low<high) {
        while(low<high&&nums[high]<=temp) {
            high--;
        }
        nums[low]=nums[high];
        while(low<high&&nums[low]>=temp) {
            low++;
        }
        nums[high]=nums[low];
    }
    nums[high]=temp;

    if(high==k-1) {
        return temp;
    } else if(high>k-1) {
        return findK(nums,k,start,high-1);
    } else {
        return findK(nums,k,high+1,end);
    }

}

int findKthLargest(int nums[],int k,int len) {
    return findK(nums,k,0,len);
}
```

```
33.
34. int main() {
35.     int nums[]= {97,76,99,102,3,5,888};
36.     cout<<findKthLargest(nums,6,7)<<endl;
37.     return 0;
38. }
```