

Quick Select(快速选择)

问题： 给出一个数组和一个数字k， 要求找出数组中第k小的元素。

1. 解法一

对这个数组排序， 然后返回arr[k-1];

对于这种解法牵扯的就是排序算法， 因此时间复杂度是 $O(n\log n)$ 。

还记得第一次投简历问的就是这个问题， 当时就是这样回答的， 然后就拜拜了。

2. 解法二

利用快速排序的思想和迭代方法进行快速切分然后找出目标值；

快排思想在于每一次确定pivot在有序数组中的位置， 然后递归调用左边和右边。因此在这里也是一样， 我们可以通过一次次的确定pivot的位置不断缩小数组可能范围， 最终确定第k小元素。

代码：

```
1. public class QuickSelect {
2.
3.     public static int quickSelect(int[] A, int k) {
4.         int length = A.length;
5.         return helper(A, 0, length - 1, k);
6.     }
7.
8.     public static int helper(int[] A, int left, int right, int k) {
9.         while (left <= right) {
10.             int partition = partition(A, left, right);
11.             System.out.println(partition);
12.             int index = partition - left + 1;
13.             if (index == k)
14.                 return A[partition];
15.             else if (k < index) {
16.                 right = partition - 1;
17.             } else {
18.                 left = partition + 1;
19.                 k = k - index;
20.             }
21.         }
22.         return -1;
23.     }
24.
25.     public static int partition(int[] A, int left, int right) {
26.         int pivot = A[left];
27.         int i = left + 1;
28.         int j = right;
29.         while (i < j) {
30.             while (i < right && A[i] < pivot) i++;
31.             while (j > left && A[j] > pivot) j--;
32.             if (i < j) {
33.                 swap(A, i, j);
34.             }
35.         }
36.         swap(A, left, j);
37.         return j;
38.     }
39.
40.     public static void swap(int[] A, int i, int j) {
41.         int temp = A[i];
42.         A[i] = A[j];
43.         A[j] = temp;
44.     }
45.
46.     public static void main(String[] args) {
47.         int[] A = {6,10,13,5,8,3,2,11};
48.         System.out.println(quickSelect(A, 8));
49.     }
50.
51. }
52. 该代码在eclipse上测试通过。
```

时间复杂度 $O(n)$