7.3.1 快速排序

7.3.1 快速排序

1 快速排序的定义和优点

快速排序是冒泡排序的改进，在排序过程中数据移动少，是目前已知的最快的排序方法。

2 快速排序的基本思想

基本思想：划分和分治递归。
第一步划分：将整个数组划分为两个部分，第一部分所有值小于基准值（key），第二部分所以值大于基准值（key）。基准值的选择是随机的，（一般选择待排数组的第一个元素）。
第二步分治递归：第一步将数组划分为两部分后，两部分内部还不是有序的，再分别对两部分递归地进行快速排序，最终得到一个完整的有序数组。

7.3.1 快速排序

3 快速排序的流程

(1) left、right 指针 (索引) 分别指向待排数组的首、尾。
(2) left 指针向后遍历，right 指针向前遍历。
(3) 当 right 指针指向元素小于基准值 (key) 时，right 指针元素便赋值给 left 指针元素，完成转移。
(4) 当 left 指针指向元素大于基准值 (key) 时，left 指针元素便赋值给 right 指针元素，完成转移。
(5) 最终当 left=right 时，遍历结束。
(6) 以基准值 (key) 为界限，把数组分成两部分，分别对这两部分进行快速排序 (显然这是一个递归的过程)

7.3.1 快速排序

4 快速排序的举例

7.3.1 快速排序

5 快速排序的小技巧

(1) 若以数组的第一个元素为基准值，则应该先用 right 向左遍历，然后再用 left 向右遍历。

(2) 若以数组的最后一个元素为基准值，则应该先用 left 向右遍历，然后再用 right 向左遍历。

(3) 一般不建议用数组中间的元素做基准值，但也可以用。

7.3.1 快速排序

6 快速排序的源代码

```
template<typename T,size_t N>
void array<T,N>::quicksort(T a[],int left,int right)
{
  if (right <left)
  cout<<" 输入错误";
  else if(right-left==1)
   {
   if (a[left] > a[right])
   swap(a[left], a[right]);
   }
  else
   {T key = a[left];int i = left;int j = right;
    while (i != j)
      { while (i < j && a[j] >= key) j--;
        while (i < j && a[i] <= key) i++;
        if (i < j) swap(a[j], a[i]); }
            a[left] = a[j]; a[i] = key;
     quicksort(a, left, i - 1);//递归调用快速排序
     quicksort(a, i + 1, right);//递归调用快速排序
   }
}
```

7.3.1 快速排序

7 课堂小测试

对下列序列进行快速排序，都以第一个元素为基准进行第一次划分，则在该次划分过程中，需要移动元素次数最多的序列是 ()

(A) 1,3,5,7,9 (B)9,7,5,3,1

(c) 5,1,3,7,9 (D)5,7,9,3,1

问你问题要快速回答