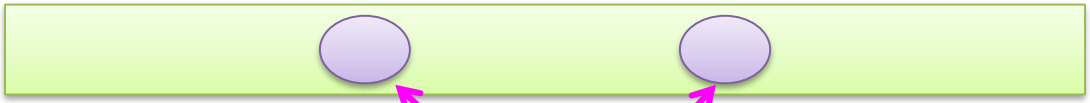
10.3 交换排序

基本思路



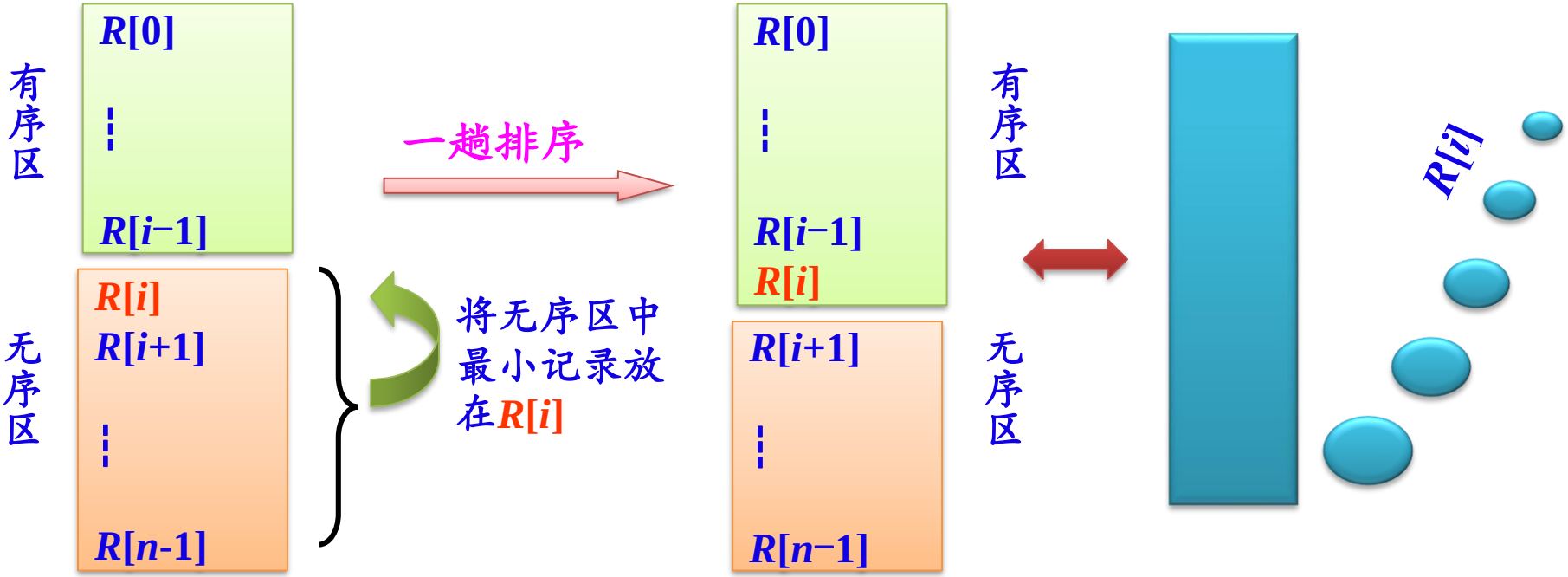
两个记录反序时进行交换

常见的交换排序方法：

（1）冒泡排序（或起泡排序）

（2）快速排序

10.3.1 冒泡排序



初始有序区为空。

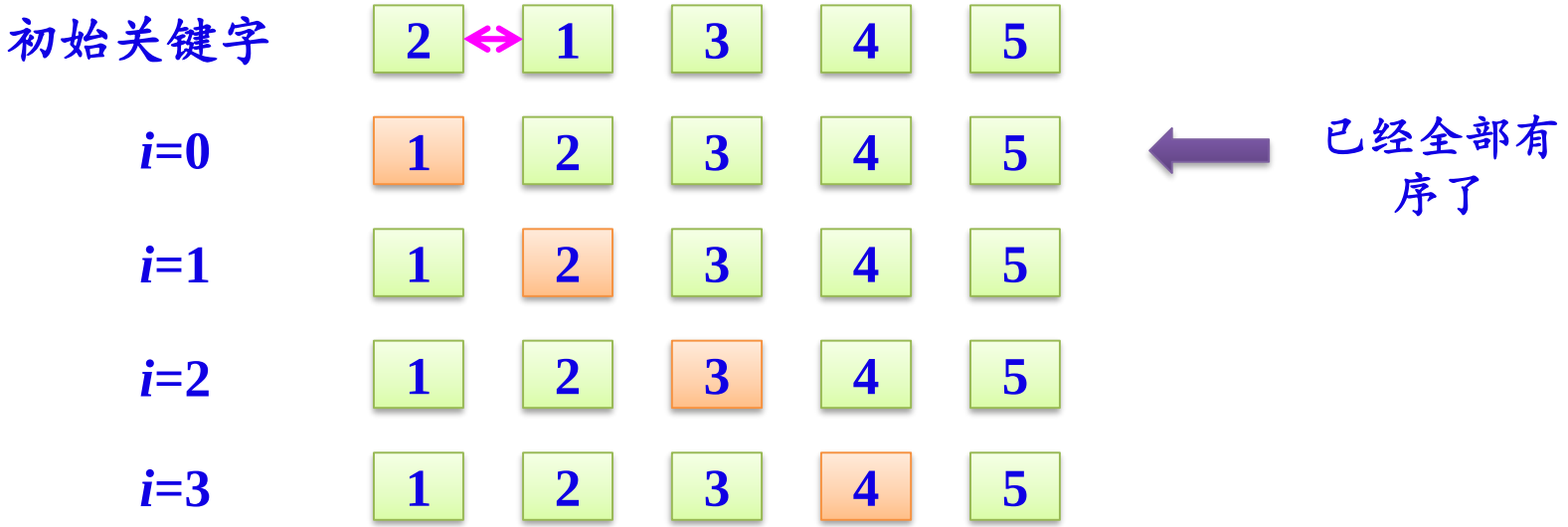
$i=0\sim n-2$，共$n-1$趟使整个数据有序。

有序区总是全局有序的

冒泡排序算法

```
void BubbleSort(RecType R[],int n)
{     int i,j;  RecType temp;
      for (i=0;i<n-1;i++)
      {
          for (j=n-1;j>i;j--)                    //比较找本趟最小关键字的记录
              if (R[j].key<R[j-1].key)
               {          temp=R[j];            //R[j]⇔R[j-1]
                          R[j]=R[j-1];
                          R[j-1]=temp;
               }
     }
}
```

采用前面的冒泡排序方法对(2,1,3,4,5) 进行排序



如何提高效率？

一旦某一趟比较时不出现记录交换，说明已排好序了，就可以结束本算法。

改进冒泡排序算法：

```
void   BubbleSort(RecType R[],int n)
{      int i, j;  bool exchange;  RecType temp;
       for (i=0;i<n-1;i++)
       {       exchange=false;
               for (j=n-1;j>i;j--)              //比较,找出最小关键字的记录
                 if (R[j].key<R[j-1].key)
                   {     temp=R[j];  R[j]=R[j-1];  R[j-1]=temp;
                         exchange=true;
                   }
              if (exchange==false) return;  //中途结束算法
       }
}
```

算法分析

最好的情况（关键字在记录序列中正序）：只需进行一趟冒泡

“比较”的次数：

$n-1$

“移动”的次数：

0

最坏的情况（关键字在记录序列中反序）：需进行$n-1$趟冒泡

“比较”的次数：

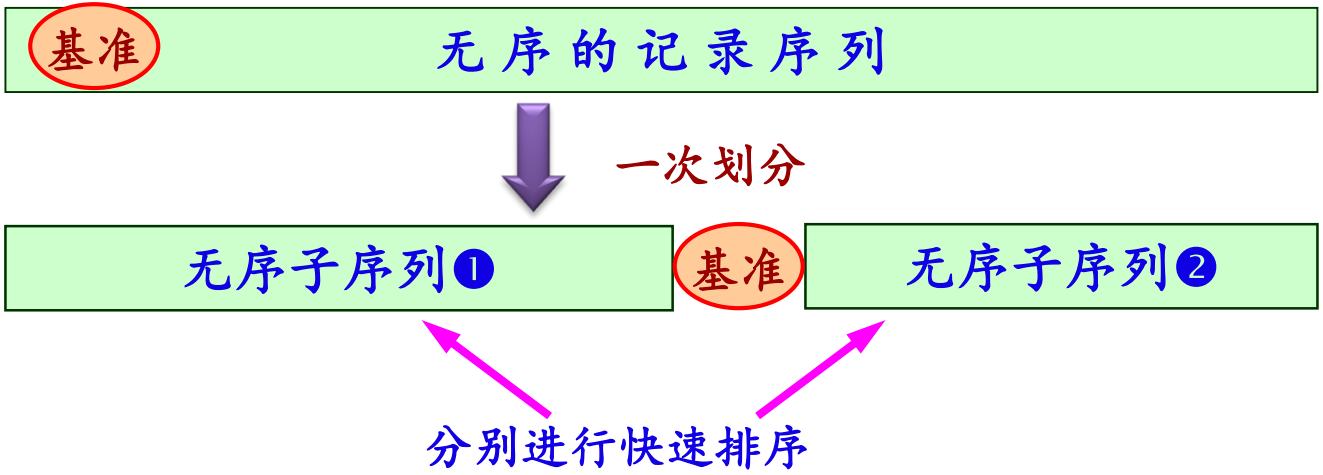
$$\sum_{i=0}^{n-1}(n-i-1)=\frac{n(n-1)}{2}$$

“移动”的次数：

$$\sum_{i=0}^{n2}3(n\ \ i-1)=\frac{3n(n-1)}{2}$$

所以冒泡排序最好时间复杂度为$O(n)$，最坏和平均为$O(n^2)$。

10.3.2 快速排序



每趟使表的第1个元素放入适当位置（归位），将表一分为二，对子表按递归方式继续这种划分，直至划分的子表长为0或1（递归出口）。

回顾划分：示例

整个区间：$R[s..t]$

tmp

$i=j$：区间处理完毕

划分完毕

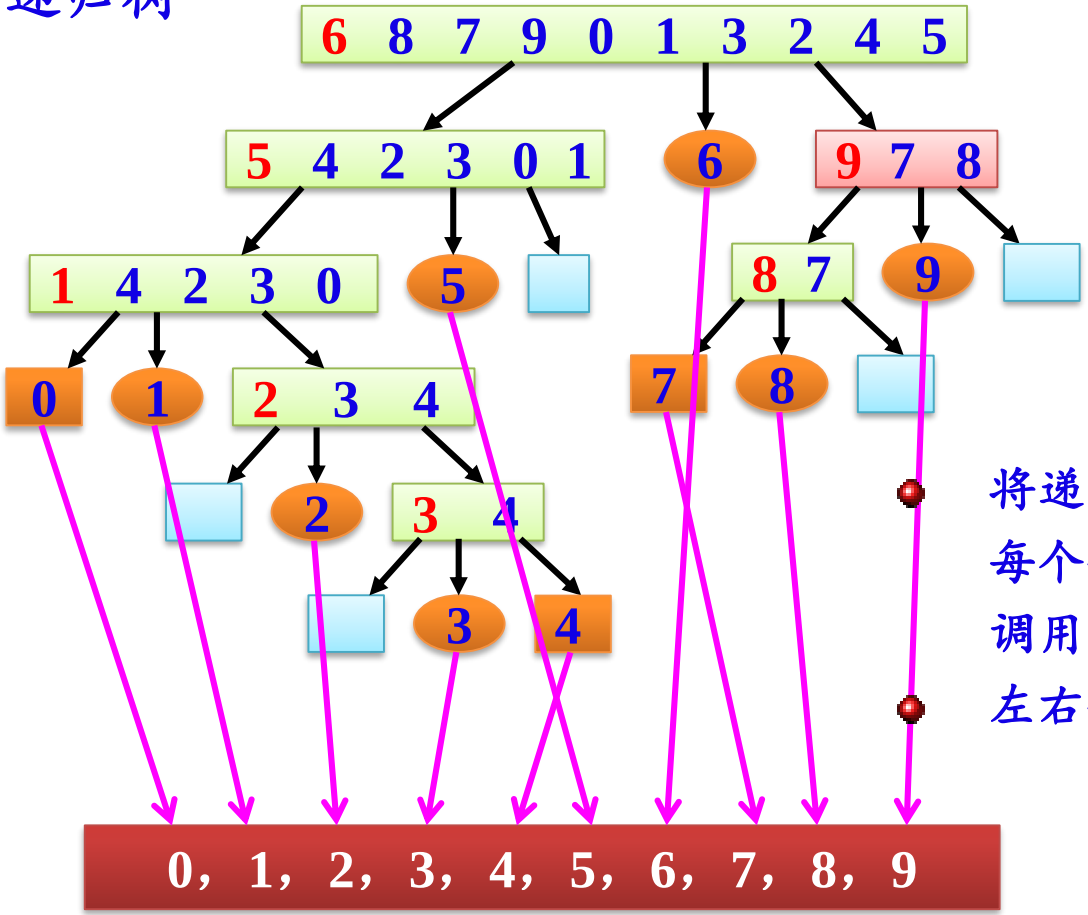
快速排序算法

```
void QuickSort(RecType R[],int s,int t)
//对R[s]至R[t]的元素进行快速排序
{     int i=s,j=t;   RecType tmp;
      if (s<t)                              //区间内至少存在2个元素的情况
      {     tmp=R[s];                       //用区间的第1个记录作为基准
            while (i!=j)                    //从两端交替向中间扫描,直至i=j为止
            {     while (j>i && R[j].key>=tmp.key)   j--;
                  R[i]=R[j];
                  while (i<j && R[i].key<=tmp.key)   i++;
                  R[j]=R[i];
            }
            R[i]=tmp;
            QuickSort(R,s,i-1);             //对左区间递归排序
            QuickSort(R,i+1,t);             //对右区间递归排序
      }
      //递归出口：不需要任何操作
}
```

一次划分

【例10-2】设待排序的表有10个记录，其关键字分别为{6，8，7，9，0，1，3，2，4，5}。说明采用快速排序方法进行排序的过程。

快速排序递归树



将递归树看成一颗3叉树，
每个分支节点对应一次递归调用。这里递归次数：7
左右分区处理的顺序无关

【例10-3】采用递归方式对顺序表进行快速排序，下列关于递归次数的叙述中，正确的是_____。

A. 递归次数与初始数据的排列次序无关

B. 每次划分后，先处理较长的分区可以减少递归次数

C. 每次划分后，先处理较短的分区可以减少递归次数

D. 递归次数与每次划分后得到的分区处理顺序无关

说明：本题为2010年全国考研题

【例10-4】为实现快速排序法，待排序序列宜采用存储方式是_____。

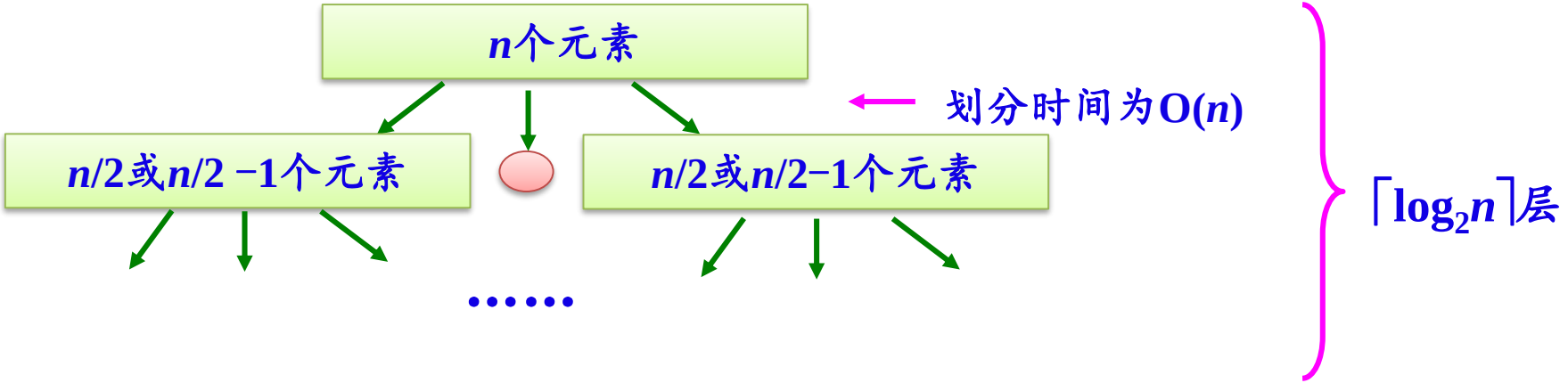
A. 顺序存储　　B. 散列存储

C. 链式存储　　D. 索引存储

说明：本题为2011年全国考研题

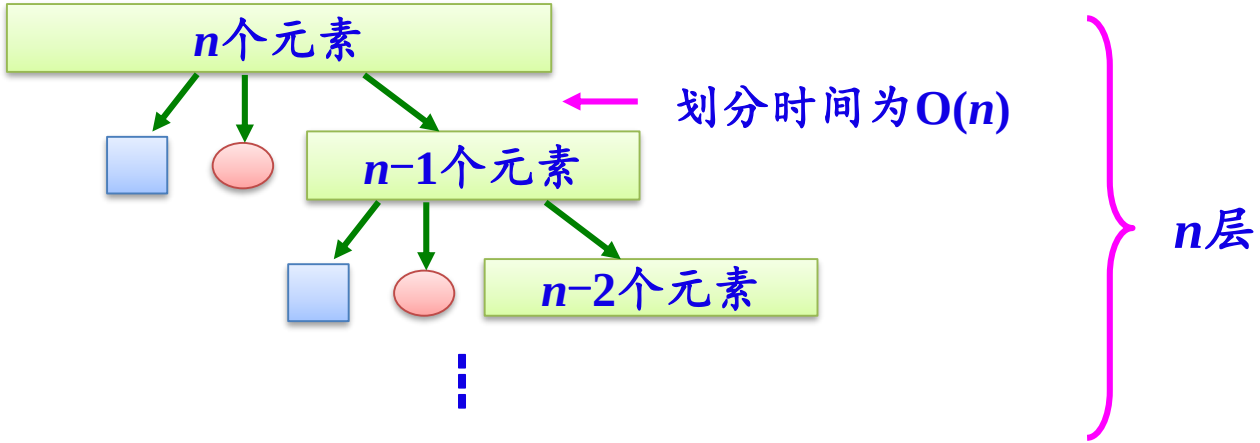
算法分析

最好情况：



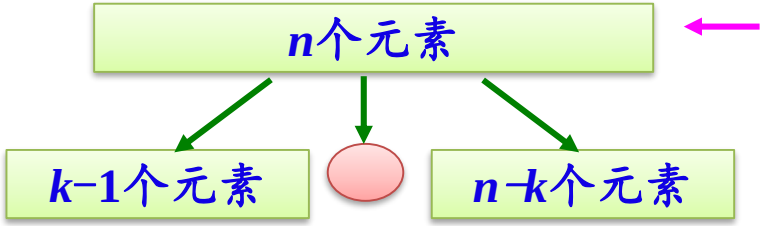
此时时间复杂度为$O(nlog_2n)$，空间复杂度为$O(log_2n)$。

最坏情况：



此时时间复杂度为$O(n^2)$，空间复杂度为$O(n)$。

平均情况：



k：1~n，共有n种情况

由此可得快速排序所需时间的平均值为：

$$T_{avg}(n) = Cn + \frac{1}{n}\sum_{k=1}^{n}\left[T_{avg}(k-1) + T_{avg}(n-k)\right]$$

1次划分的时间

则可得结果：$T_{avg}(n)=Cn\log_2 n$。

结论：快速排序的平均时间复杂度为$O(n\log_2 n)$。

平均所需栈空间为$O(\log_2 n)$。

思考题

快速排序的最坏时间复杂度为$O(n^2)$，与冒泡排序相同。为什么快速排序更好？

——本讲完——