课时十　查找

考点	重要程度	占分	题型
1.查找	★★	0~2	选择
2.顺序查找	★★★★★	4~8	选择、填空
3.折半查找	必考	4~8	选择、填空
4.散列表	★★★★★	6~10	解答

10.1查找的基本概念

（1）查找。在数据集合中，寻找满足某种条件的数据元素的过程称为查找。

（2）查找表（查找结构）。用于查找的数据集合称为查找表。

（3）关键字是数据元素中某个数据项的值，用它可以标识一个数据元素，若此关键字可 以唯一地标识一个记录，用词关键字为主关键字。

（4）平均查找长度是所有查找过程中进行关键字的比较次数的平均值。

10.2顺序查找

顺序查找，主要用于线性表中进行查找。

其基本思想是从线性表的一端开始，逐个检查关键字是否满足给定的条件。若查找到某个 元素的关键字满足给定条件，则查找成功，返回该元素在线性表中的位置。若查找到表的 另一端，仍未找到符合给定条件的元素，则返回查找失败的信息。

10.2顺序查找

```
typedef struct{ ElemType *elem;        //查找表的数据结构
int TableLen;                          //元素存储空间基址，建表时按实际长度
}SSTable;                              //分配， 号元素留空
int Search_Seq(SSTable ST,ElemType key){
ST.elem[0]=key;                        // “哨兵”
for(i=ST.TableLen;ST.elem[i]!=key;--i);
return i;
}
```

引入“哨兵”的目的是使得 Search_Seq 内的循环不必判断是否会越界，因为满足 i==0，循环一定会跳出。

10.2顺序查找

对于 n 个元素的表，给定值 key 与表中第 个元素相等，需进行 n-i+1 次关键字的比 较，查找成功时，顺序查找的平均查找长度为 $ASL_{成功}=\sum_{i=1}^{n} p_i(n-i+1)=\frac{n+1}{2}$

查找不成功时，与表中各关键字的比较次数是 NLR 次，从而顺序查找不成功的平均查找长度为 $ASL_{不成功}= n+1$。

注意：对线性链表只能进行顺序查找。

题 1.采用顺序查找方法查找长度为 的顺序表时，它的平均查找长度为（ ）。

A.n　　B.n/2　　C.(n+1)/2　　D.(n-1)/2

答案：C

10.3 折半查找

折半查找（二分查找），仅适用于有序的顺序表。

折半查找的基本思想，首先将给定值与表中中间位置的元素比较，若相等，则查找成 功，返回该元素的存储位置；若不等，则所需查找的元素只能在中间元素以外的前半部分 或后半部分。然后在缩小的范围内继续进行同样的查找，如此反复，直至找到为止，或确 定表中没有所需要查找的元素，则查找不成功，返回查找失败的信息。

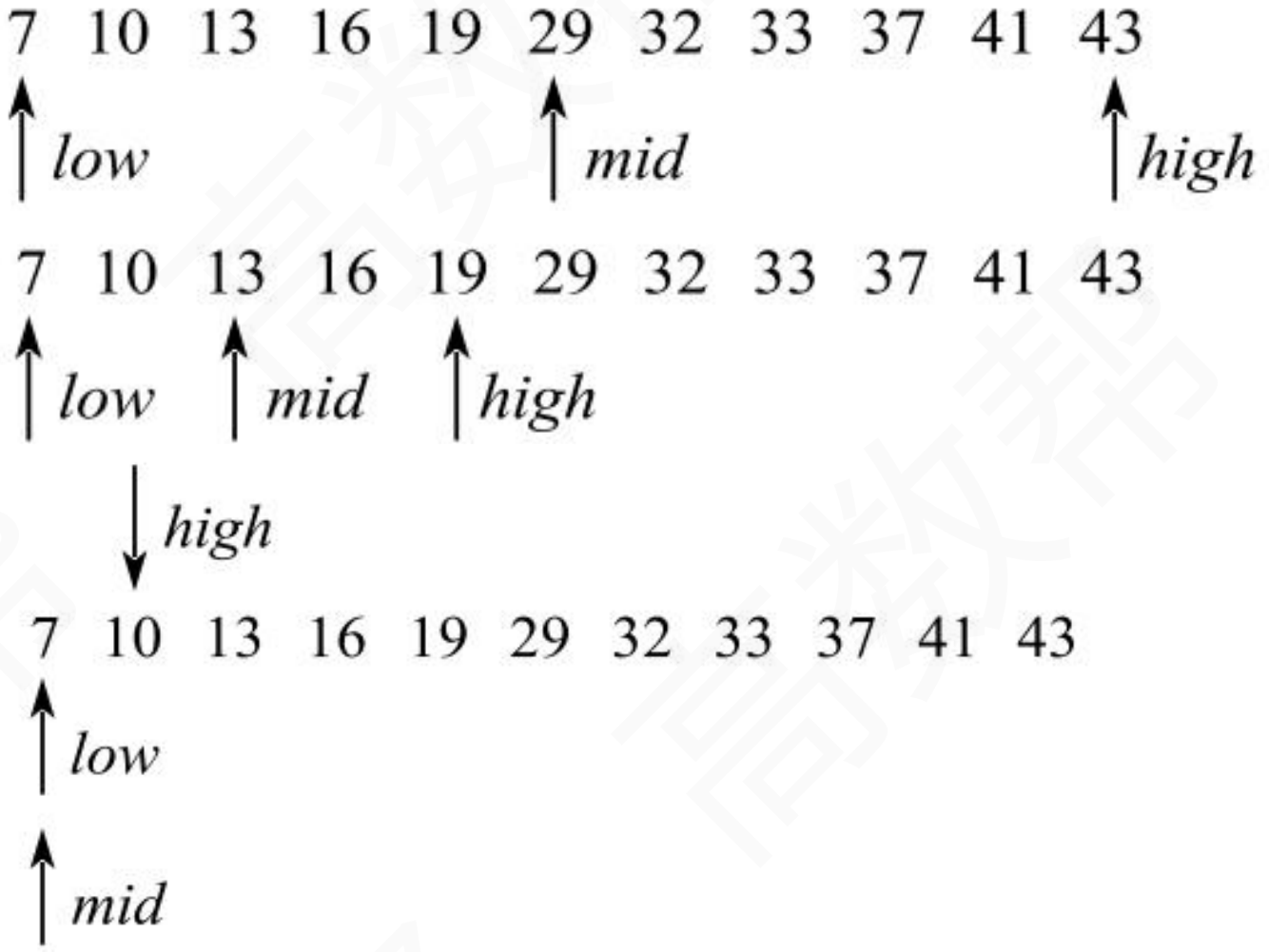
10.3 折半查找

```
int Binary_Search(SeqList L,ElemType key){
int low=0,high=L.TableLen-1,mid;
while(low<=high){
mid=(low+high)/2;
if(L.elem[mid]==key)
return mid;
else if(L.elem[mid]>key)
high=mid-1;
else
low=mid+1; }
}
```

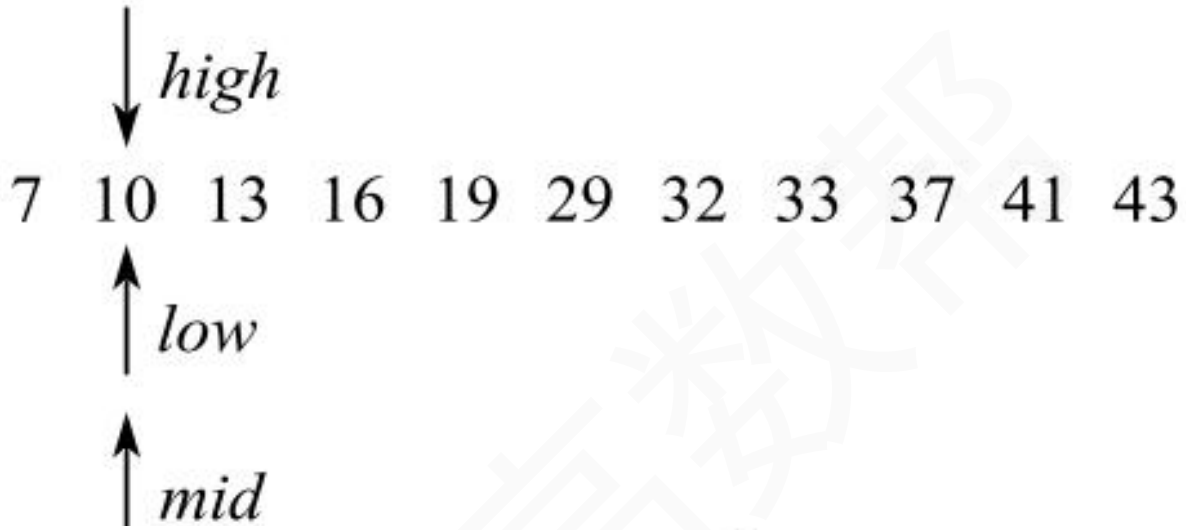
视频讲解更清晰
仅4小时

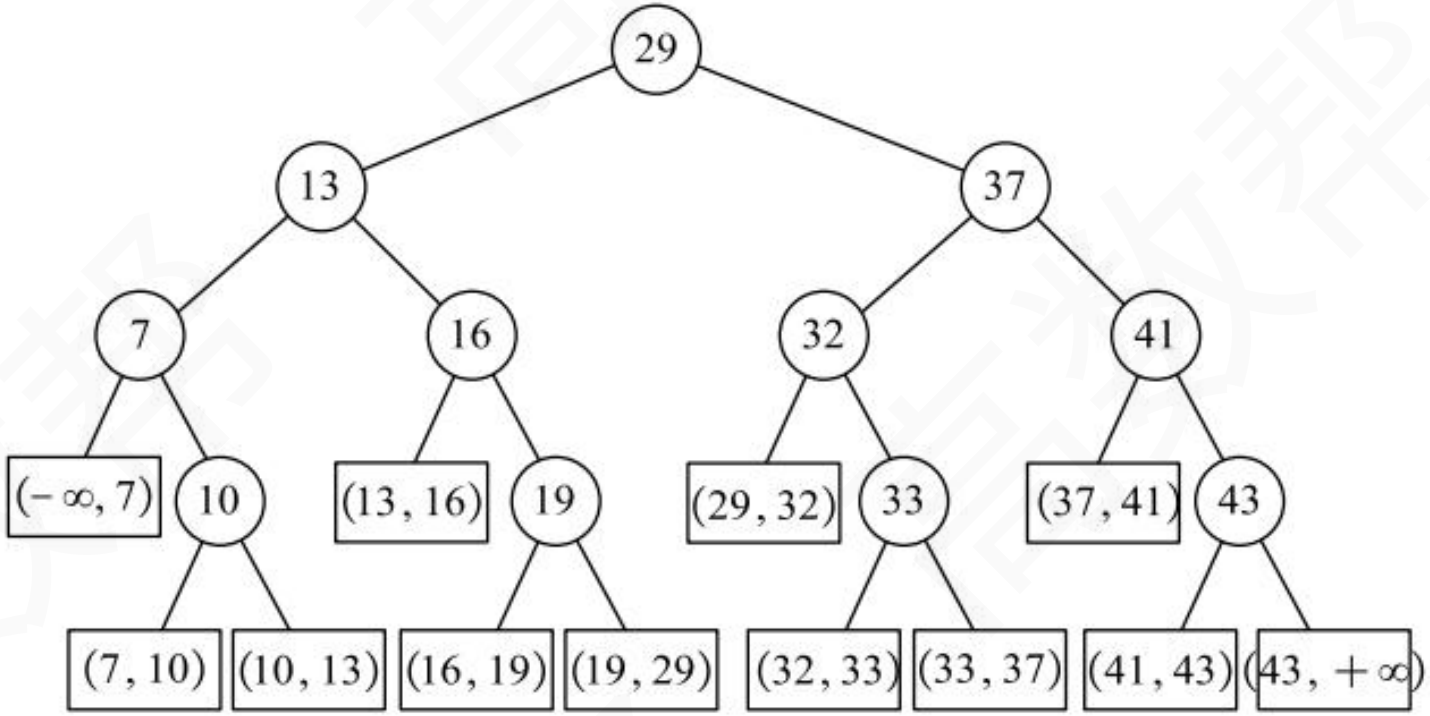
10.3 折半查找

已知 11 个元素的有序表{7，10，13，16，19，29，32，33，37，41，43}，查找值为11 的元素。



10.3 折半查找


high
7 10 13 16 19 29 32 33 37 41 43
low
mid


29
13
37
7
16
32
41
(−∞, 7)
10
(13, 16)
19
(29, 32)
33
(37, 41)
43
(7, 10)
(10, 13)
(16, 19)
(19, 29)
(32, 33)
(33, 37)
(41, 43)
(43, +∞)

10.3 折半查找

在等概率查找时，查找成功的平均查找长度是

$$ASL=\frac{1}{n}\sum_{i=1}^{n}l_i=\frac{1}{n}(1\times1+2\times2+\cdots+h\times2^{h-1})=\frac{n+1}{n}\log_2(n+1)-1\approx\log_2(n+1)-1$$

其中，h 是树的高度，并且元素个数为 n 时树高$h=[\log_2(n+1)]$。所以，

折半查找的时 间复杂度为 。 折半查找法仅适用于顺序存储结构，不适用于链式存储结构，且要求元素按关键字有序 排列。

题 1.二分查找要求结点（ ）。

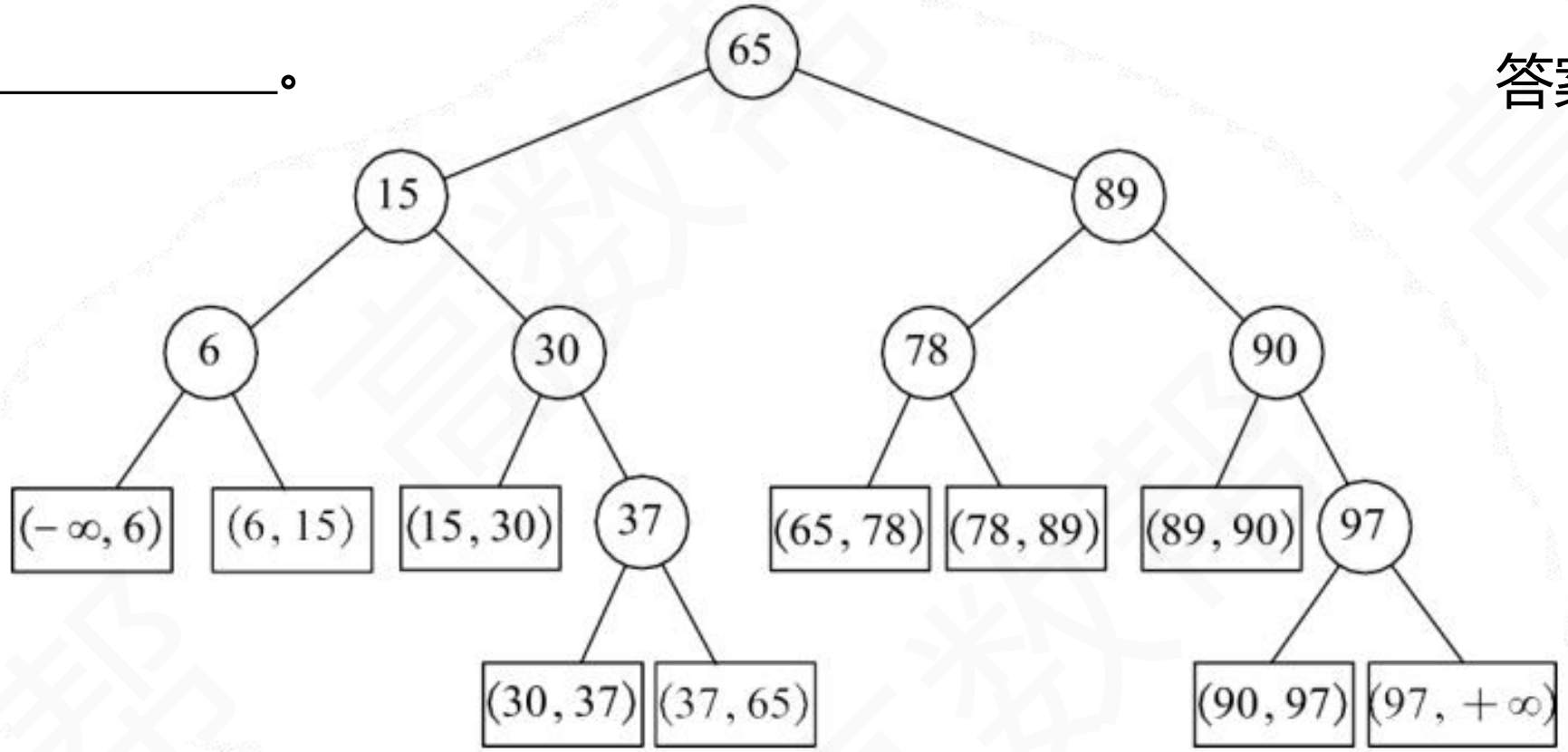
A.有序、顺序存储　　B.有序、链式存储

C.无序、顺序存储　　D.无序、链式存储

答案：A

10.3 折半查找

题 2.采用对分查找序列数据(6,15,30,37,65,78,89,90,97)，若查找元素 时比较次数为 _____。



答案：2

题3.采用二分查找方法查找长度为n的线性表时，每个元素的平均查找长度为____。

A．O（n2）　　B. O(nlog2n)　　C. O(n)　　D. O(log2n)

答案：D

10.3 折半查找

常见的散列函数：

（1）直接定址法：直接取关键字的某个线性函数值为散列地址，散列函数为

$$H(key)=key \text{ 或 } H(key)=a\times key+b \text{（a,b 是常数）}$$

（2）除留余数法：假定散列表表长为 m 取一个不大于 但最接近或等于 m 的质数 p，利用以下公式把关键字转换成散列地址。

$$H(key)=key\%p$$

（3）数字分析法

（4）平方取中法

10.3 折半查找

解决冲突的方法：

（1）开放定址法：可存放新表项的空闲地址既向它的同义词表项开放，又向它的非同义 词表项开放。其数学递推公式为 $H_i=(H(key)+d_i)\%m(H(key)$（ 为散列函数，m表示散列表表长， d_i为增量序列）。

对增量序列通常有以下取法：

① 线性探测法。当$d_i=0,1,2,\cdots,m-1$时，称为线性探测法。这种方法的特点是： 冲突发生时，顺序查看表中下一个单元（探测到表尾地址$m-1$时，下一个探测 地址就是表首地址 ），直到找出一个空闲单元或查遍全表。

② 平方探测法。当$d_i=0^2,1^2,-1^2,2^2,-2^2,\cdots,k^2,-k^2$时，称为平方探测法，其中$k\leqslant m/2$。

③ 再散列法。当$d_i=Hash_2(key)$ 时，称为再散列法。

10.3 折半查找

题 1.关键字序列为 $\{19, 14, 23, 01, 68, 20, 84, 27, 55, 11, 10, 79\}$，按散列函数 H(key)=key%13 和线性探测处理冲突构造散列表，其中散列表长度为 16。

答案：$H(19) = 19\%13 = 6$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
关键字							19									
比较次数							1									

$H(14) = 14\%13 = 1$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
关键字		14					19									
比较次数		1					1									

10.3 折半查找

$H(23)=23\%13=10$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
关键字		14					19				23					
比较次数		1					1				1					

$H(01)=01\%13=1$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
关键字		14	01				19				23					
比较次数		1	2				1				1					

$H(68)=68\%13=3$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
关键字		14	01	68			19				23					
比较次数		1	2	1			1				1					

10.3 折半查找

$H(20)=20\%13=7$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
关键字		14	01	68			19	20			23					
比较次数		1	2	1			1	1			1					

$H(84)=84\%13=6$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
关键字		14	01	68			19	20	84		23					
比较次数		1	2	1			1	1	3		1					

$H(27)=27\%13=1$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
关键字		14	01	68	27		19	20	84		23					
比较次数		1	2	1	4		1	1	3		1					

10.3 折半查找

$H(55)=55\%13=3$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
关键字		14	01	68	27	55	19	20	84		23					
比较次数		1	2	1	4	3	1	1	3		1					

$H(11)=11\%13=11$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
关键字		14	01	68	27	55	19	20	84		23	11				
比较次数		1	2	1	4	3	1	1	3		1	1				

$H(10)=10\%13=10$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
关键字		14	01	68	27	55	19	20	84		23	11	10			
比较次数		1	2	1	4	3	1	1	3		1	1	3			

10.3 折半查找

$H(79)=79\%13=1$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
关键字		14	01	68	27	55	19	20	84	79	23	11	10			
比较次数		1	2	1	4	3	1	1	3	9	1	1	3			

查找成功的平均查找长度 $ASL_{成功}$=查找次数/元素个数=$(1\times6+2+3\times3+4+9)/12=2.5$

查找不成功的平均查找长度为 不成功=查找次数/散列后的地址个数=

$$(1+13+12+11+10+9+8+7+6+5+4+3+2)/13=7$$

散列表的查找长度取决于三个因素：散列函数、处理冲突的方法和装填因子。

装填因子a= 表中记录数n /散列表长度m 。

散列表的平均查找长度依赖于散列表的装填因子 a，不直接依赖于 n 或 m。

a 越大，表示装填的记录越“满”，发生冲突的可能性就越大。

（2）拉链法：为了避免非同义词发生冲突，把所有的同义词存储在一个线性链表中。

题2.设散列表容量为7（散列地址空间0..6），给定表（30，36，47，52，34）,散列函数H（K）=K mod 6，采用线性探测法解决冲突，要求：

（1）构造散列表；

（2）求查找数34需要比较的次数。

视频讲解更清晰
仅4小时

答案：(1)表形态：

0	1	2	3	4	5	6	
30	26			52	47	34	
1	2			1	1	3	

(2)查找34 的比较次数：3

课时十一　排序（1）

考点	重要程度	占分	题型
1.排序的基本概念	★★★	2~4	选择
2.插入排序	★★★★	6~10	解答
3.交换排序	★★★★★	6~10	

11.1 排序的基本概念

排序，就是重新排列表中的元素，使表中的元素满足按关键字有序的过程。算法的稳定性。若待排序表中有两个元素 R_i 和 R_j ，其对应的关键字相同即 $Key_i=Key_j$，且在排序前 R_i 在 R_j 的前面，若使用某一排序算法排序后， R_i 仍然在 R_j 的前面，则称这个排序算法是稳定的，否则称排序算法是不稳定的。

排序：①内部排序：指在排序期间元素全部存在内存中的排序

②外部排序：是指在排序期间元素无法全部同时存放在内存中，必须在排序的 过程中根据要求不断地在内，外存之间移动的排序。

内部排序算法在执行过程中都要进行两种操作：比较和移动。

通常可以将内部排序算法分为插入排序、交换排序、选择排序、归并排序和基数排序 五大类。（基数排序不基于比较）

11.1 排序的基本概念

题 1.内部排序方法的稳定性是指该排序算法不允许有相同的关键字记录。 ()

答案：错误

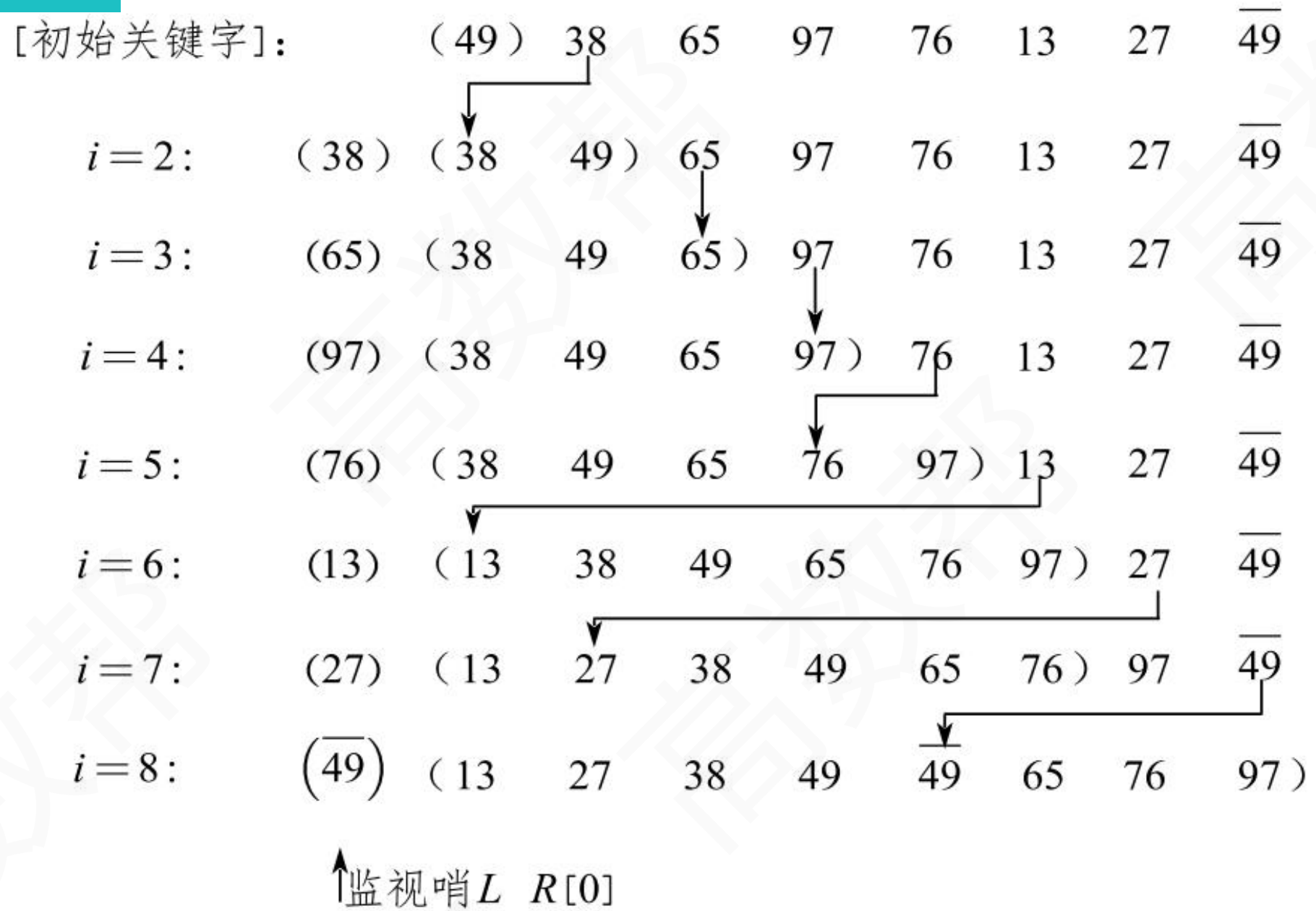
11.2 插入排序

基本思想是每次将一个待排序的记录按其关键字大小插入到前面已排好序的子序列中， 直到全部记录插入完成。

（1）直接插入排序

假定初始序列为 $49, 38, 65, 97, 76, 13, 27, \overline{49}$ ，初始时 49 可以视为一个已排好序的 子序列，按照上述算法进行直接插入排序的过程如图所示，括号内是已排好序的子序列。

11.2 插入排序


[初始关键字]: （49） 38 65 97 76 13 27 $\overline{49}$
$i=2$: （38） （38 49） 65 97 76 13 27 $\overline{49}$
$i=3$: (65) （38 49 65） 97 76 13 27 $\overline{49}$
$i=4$: (97) （38 49 65 97） 76 13 27 $\overline{49}$
$i=5$: (76) （38 49 65 76 97） 13 27 $\overline{49}$
$i=6$: (13) （13 38 49 65 76 97） 27 $\overline{49}$
$i=7$: (27) （13 27 38 49 65 76） 97 $\overline{49}$
$i=8$: ($\overline{49}$) （13 27 38 49 $\overline{49}$ 65 76 97）
监视哨 L $R[0]$

11.2 插入排序

下面是直接插入排序的代码，其中再次用到了我们前面提到的“哨兵”（作用相同）。

```
void InsertSort(ElemType A[],int n){
int i,j;  for(i=2;i<=n;i++)                    //依次将 插入到前面已排序序列
if(A[i]<A[i-1]){                               // 关键码小于前驱，将 插入有序表
A[0]=A[i];                                     //复制为哨兵， 不存放元素
for(j=i-1;A[0]<A[j];--j)                       //从后往前查找待插入位置
A[j+1]=A[j];                                   //向后挪位
A[j+1]=A[0];                                   //复制到插入位置
}
}
```

11.2 插入排序

空间效率：仅使用了常数个辅助单元，因而空间复杂度为O(1) 。

时间效率：在排序过程中，向有序子表中逐个地插入元素的操作进行了 n-1 趟，每趟操作都分为比较关键字和移动元素，而比较次数和移动次数取决于待排序表的初始状 态。直接插入排序算法的时间复杂度为 $O(n^2)$。

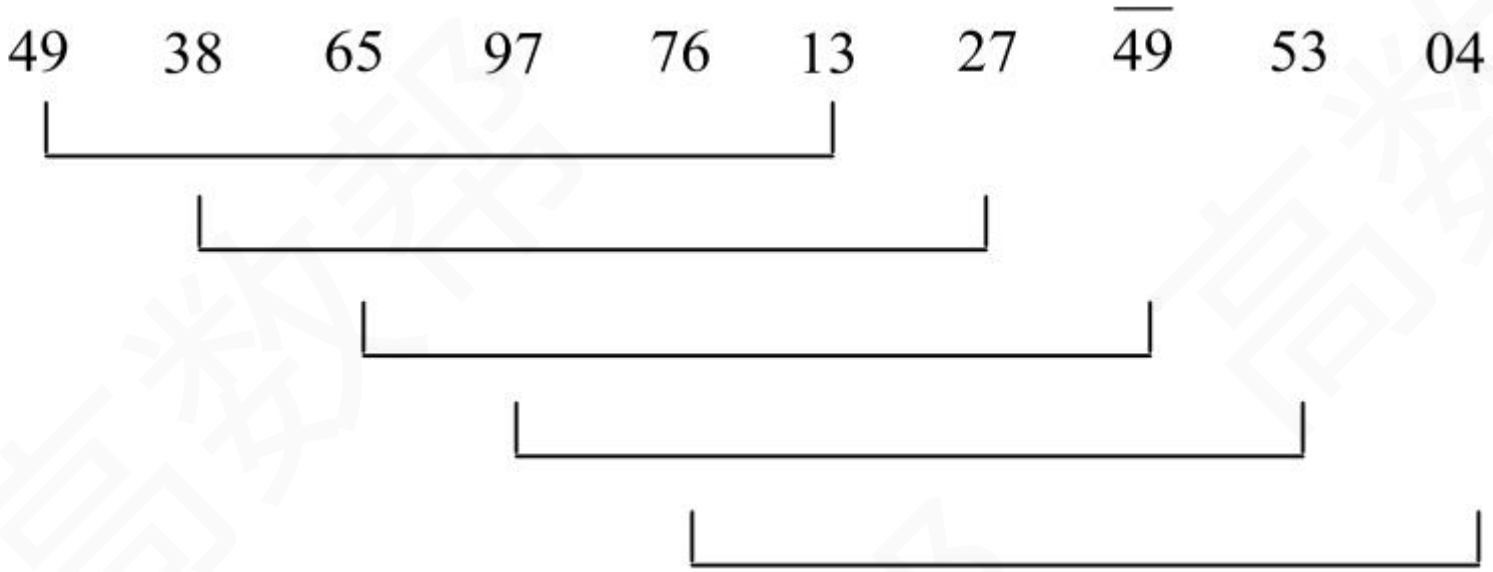
稳定性：由于每次插入元素时总是从后向前先比较再移动，所以不会出现相同元素相 对位置发生变化的情况，即直接插入排序是一个稳定的排序方法。

（2）希尔排序

先将待排序表分割成若干形如 $L[i,i+d,i+2d,\cdots i+kd]$的特殊子表，即把相隔某个增量的记录组成一个子表，对各个子表分别进行直接插入排序，当整个表中的元素已呈基 本有序时，再对全体记录进行一次直接插入排序。

11.2 插入排序

[初始关键字]：



$d=5$

一趟排序结果

$d=3$

11.2 插入排序

两趟排序结果	13	04	$\overline{49}$	38	27	49	53	65	97	76
$d=1$										
三趟排序结果	04	13	27	38	$\overline{49}$	49	53	65	76	97

空间效率：仅使用了常数个辅助单元，因而空间复杂度为 。

时间效率：由于希尔排序的时间复杂度依赖于增量序列的函数，所以其时间复杂度分 析比较困难。

稳定性：不稳定。

11.2 插入排序

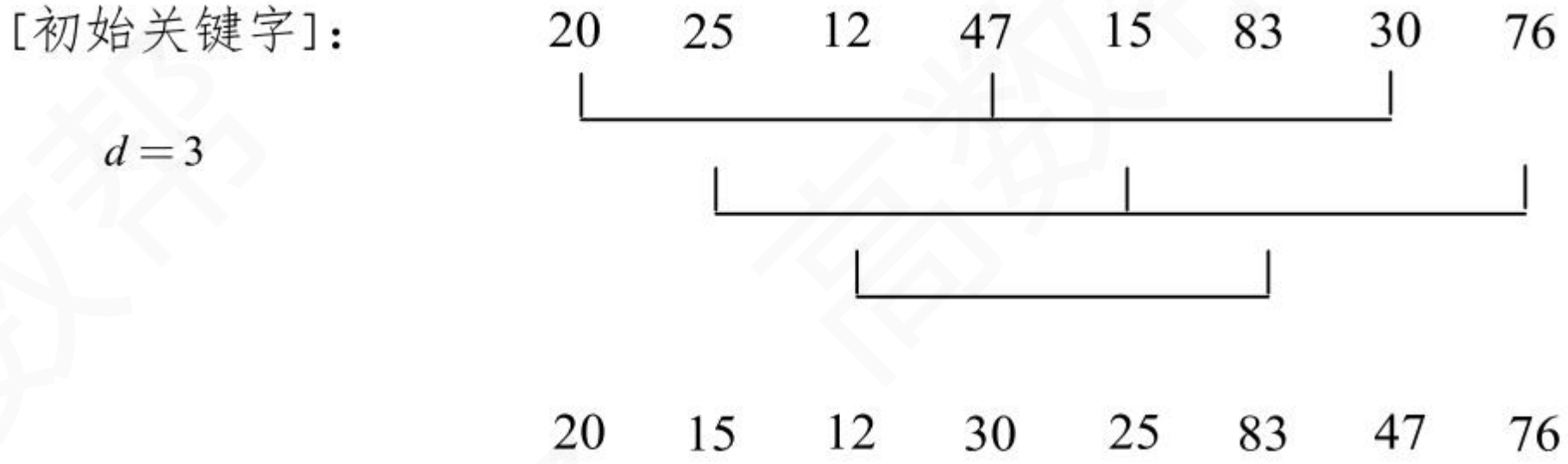
题 1.（ ）方法是从未排序的序列中挑选元素，并将其放入已排序序列的一种。

A.归并排序　　B,插入排序　　C.快速排序　　D.选择排序

答案：B

题 2.用希尔排序法对关键字序列 {20，25，12，47，15，83，30，76}进行排序时，增量为 3 的一趟 排序结果是 。

答案：20 15 12 30 25 83 47 76



11.3 交换排序

交换，是指根据序列中两个元素关键字的比较结果来对换这两个记录在序列中的位置

（1）冒泡排序

基本思想是：从后往前（或者从前往后）两两比较相邻元素的值，若为逆序，则进行交换，直到序列比较完，第一趟冒泡结束，结果是将最小的元素交换到待排序列的第一个位置（或将最大的元素交换到待排序列的最后一个位置）。下一趟冒泡时，前一趟确定的 最小元素不再参与比较，每趟冒泡的结果是把序列中的最小元素（或最大元素）放到了序列的最终位置。这样经过 n-1 趟冒泡后就能把所有元素排好序。

11.3 交换排序

初始状态	49	38	65	97	76	13	27	$\overline{49}$
第一趟后	[13]	49	38	65	97	76	27	$\overline{49}$
第二趟后	[13	27]	49	38	65	97	76	$\overline{49}$
第三趟后	[13	27	38]	49	$\overline{49}$	65	97	76
第四趟后	[13	27	38	49]	$\overline{49}$	65	76	97
第五趟后	[13	27	38	49	$\overline{49}$]	65	76	97
第六趟后	[13	27	38	49	$\overline{49}$	65]	76	97
第七趟后	[13	27	38	49	$\overline{49}$	65	76]	97

11.3 交换排序

冒泡排序算法的代码如下：

```
void BubbleSort(ElemType A[],int n){
        for(i=0;i<n-1;i++)
        flag=false;
        for(j=n-1;j>i;j--)
                if(A[j-1]>A[j]){
                        swap(A[j-1],A[j]);
                        flag=true; }
                if(flag==false)
                    return; }
```

视频讲解更清晰
仅4小时

空间效率：仅使用了常数个辅助单元，因而空间复杂度为 $O(1)$。

时间效率：时间复杂度为 $O(n^2)$ 。

稳定性：稳定。

11.3 交换排序

题 1.对 个不同的关键字由小到大进行冒泡排序，在下列（ ）情况下交换的次数最多。

A.从小到大排列好的　　B.从大到小排列好的

C.元素无序　　D.元素基本有序　　答案：B

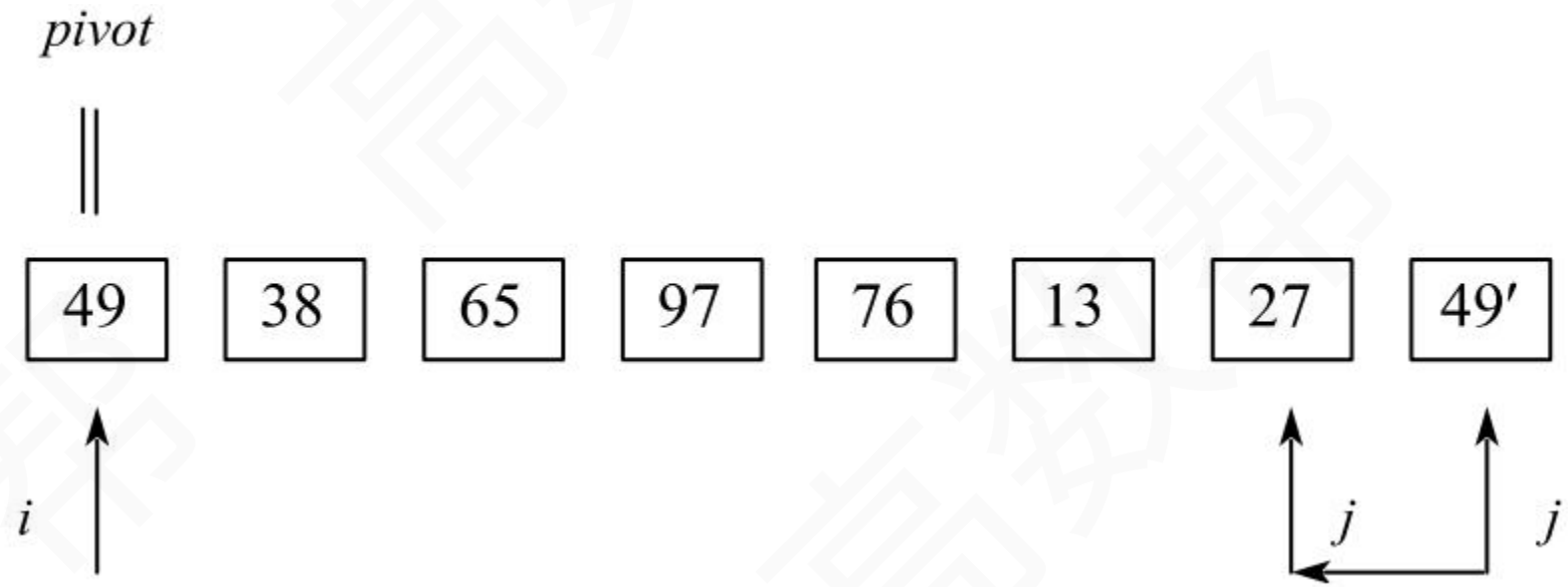
（2）快速排序

快速排序的基本思想是：在待排序表 $L[1\cdots n]$ 中任取一个元素 $pivot$ 作为枢轴（通常取首元素），通过一趟排序将待排序表划分为独立的两部分 $L[1\cdots k-1]$ 和 $L[k+1\cdots n]$，使得 $L[1\cdots k-1]$ 中的所有元素小于 $pivot$，$L[k+1\cdots n]$ 中的所有元素大于 $pivot$，则 $pivot$ 放在了最终位置 $L[k]$ 上，这个过程称为一趟快速排序。然后分别递归地对两个子表重复上述过 程，直至每部分内只有一个元素或空为止。

11.3 交换排序

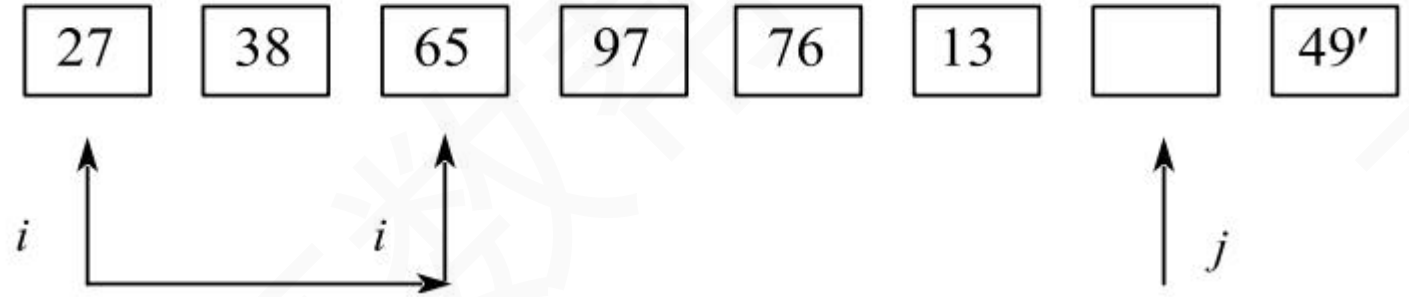
题 1.对关键字序列 $\{49, 38, 65, 97, 76, 13, 27, 49'\}$ 进行快速排序。

设两个指针 i 和 j，初值分别为 low 和 high，取第一个元素 49 为数轴赋值到变量 pivot。 指针 j从 high往前搜索找到第一个小于枢轴的元素 27，将 27 交换到 i 所指位置。

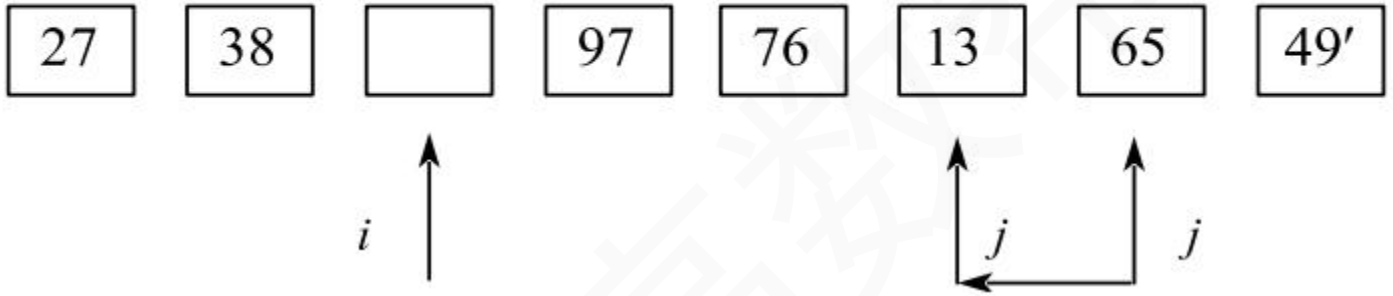


11.3 交换排序

指针i从low往后搜索找到第一个大于枢轴的元素65，将65交换到j所指位置。

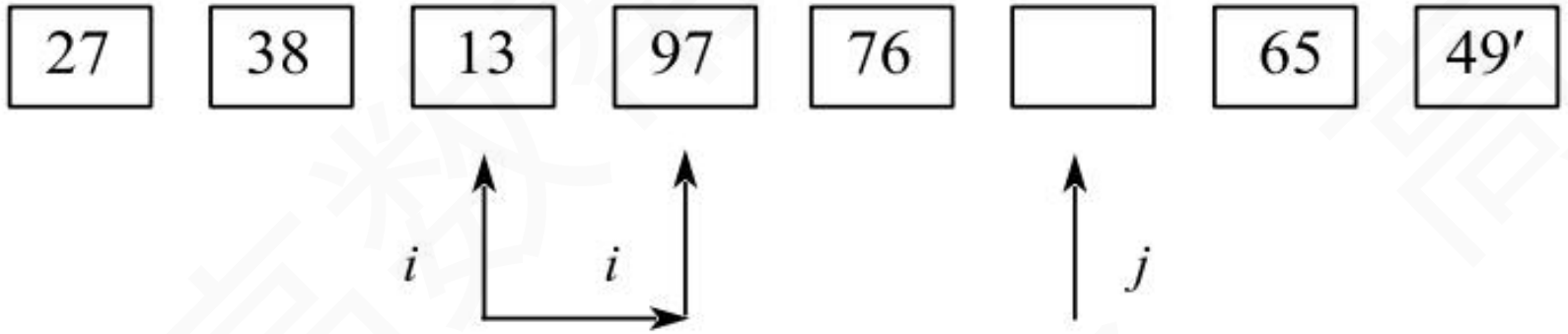


指针j继续往前搜索找到小于枢轴的元素13，将13交换到i所指位置。

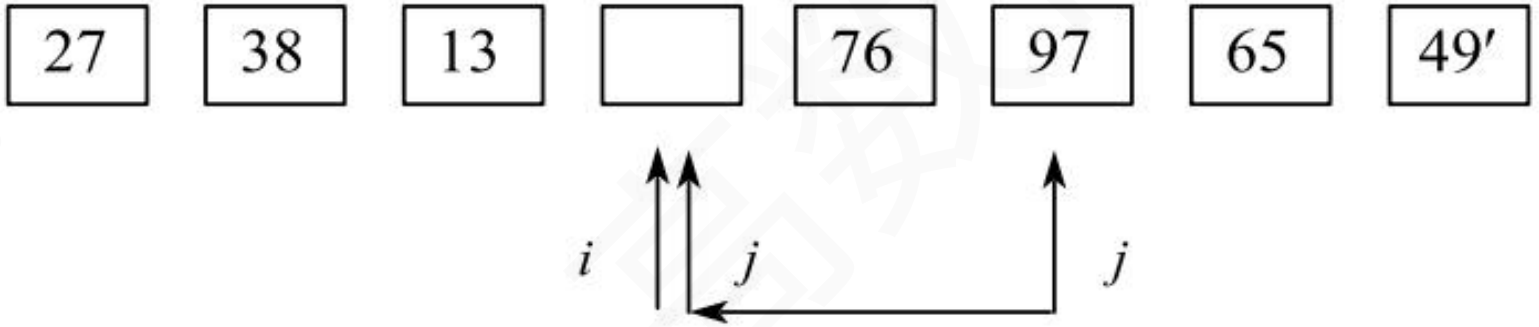


11.3 交换排序

指针i继续往后搜索找到大于枢轴的元素97，将97交换到j所指位置。



指针j继续往前搜索小于枢轴的元素，直到$i==j$。



此时，经过一次划分，将原序列分割成了前后两个子序列。

{27　38　13}　49　{76　97　65　49′}

按照同样的方法对各子序列进行快速排序，若待排序中只有一个元素，显然已有序。

{27　38　13}　49　{76　97　65　49′}

{13}　27　{38}　{49′　65}　76　{97}

49′　{65}

最后得到序列 13　27　38　49　49′　65　76　97

11.3 交换排序

空间效率：由于快速排序是递归的，需要借助栈来保存每层递归调用的必要信息，其 容量要与递归调用的最大深度一致，因而空间复杂度为 $O(\log_2 n)$。

时间效率：时间复杂度为 $O(n \log_2 n)$。

稳定性：不稳定。

快速排序是所有内部排序算法中平均性能最优的排序算法。 快速排序并不适用于原本有序或基本有序的记录序列进行排序。

课时十二　排序（2）

考点	重要程度	占分	题型
1.选择排序	★★★★★	6~10	选择、填空、解答
2.归并排序	★★★★	4~6	
3.基数排序	★★★★	4~6	
4.各种排序算法的比较	★★★★	2~4	选择

12.1 选择排序

选择排序的基本思想：

每一趟（第 i 趟）在后面$n-i+1(i=1,2,\cdots,n-1)$个待排序元素中选取关键字最小的元 素，作为有序子序列的第 i 个元素， 直到$n-1$趟做完，待排序元素只剩下一个，就不用再选了。

（1）简单选择排序

基本思想：假设排序表为$L[1\cdots n]$，第 i 趟排序即从$L[i\cdots n]$中选择关键字最小的元素与$L[i]$交换，每一趟排序可以确定一个元素的最终位置，这样经过n-1趟排序就可使得 整个排序表有序。

12.1 选择排序

简单选择排序算法的代码如下：

```
void SelectSort(ElemType A[],int n){
for(i=0;i<n-1;i++){
min=i;
for(j=i+1;j<n;j++)
if(A[j]<A[min])
min=j;
if(min!=i)
swap(A[i],A[min]);   }
}
```

12.1 选择排序

空间效率：$O(1)$

时间效率：$O(n^2)$

稳定性：不稳定。

例如，表 $L=\{2, 2', 1\}$，经过一趟排序后 $L=\{1, 2', 2\}$，

最终排序序列也是 $L=\{1, 2', 2\}$

题 1.给定关键字序列 $\{87, 45, 78, 32, 17, 65, 53, 09\}$，用简单选择排序算法进行排序。

	87	45	78	32	17	65	53	09
第一趟排序	[09]	45	78	32	17	65	53	87
第二趟排序	[09	17]	78	32	45	65	53	87
第三趟排序	[09	17	32]	78	45	65	53	87
第四趟排序	[09	17	32	45]	78	65	53	87
第五趟排序	[09	17	32	45	53]	65	78	87
第六趟排序	[09	17	32	45	53	65]	78	87
第七趟排序	[09	17	32	45	53	65	78]	87

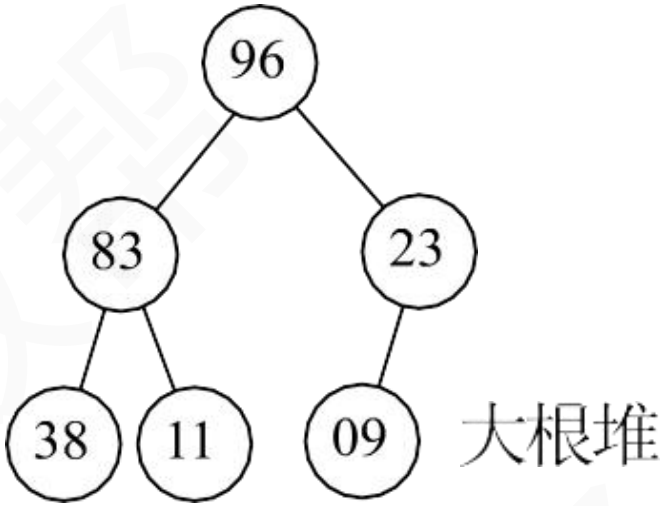
（2）堆排序

堆的定义如下，n 个关键字序列$L[1\cdots n]$称为堆，当且仅当该序列满足：

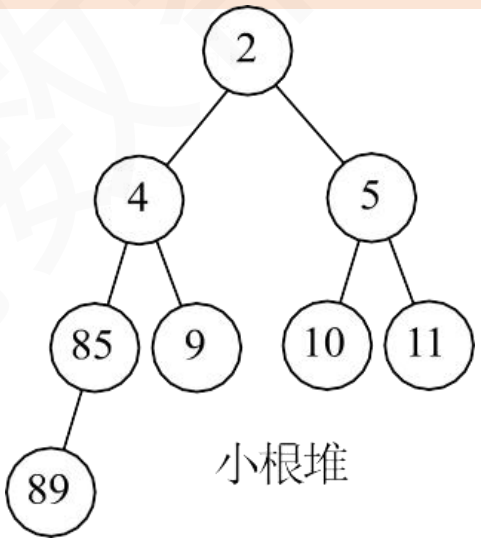
① $L[i]>=L[2i]$且 $L[i]>=L[2i+1]$

② $L[i]<=L[2i]$且 $L[i]<=L[2i+1]$ $1\leqslant i\leqslant\lfloor n/2\rfloor$

满足条件①的堆称为大根堆，大根堆的最大元素存放在根结点，且其任一非根结点的 值小于等于其双亲结点值。满足条件②的堆称为小根堆。



大根堆



小根堆

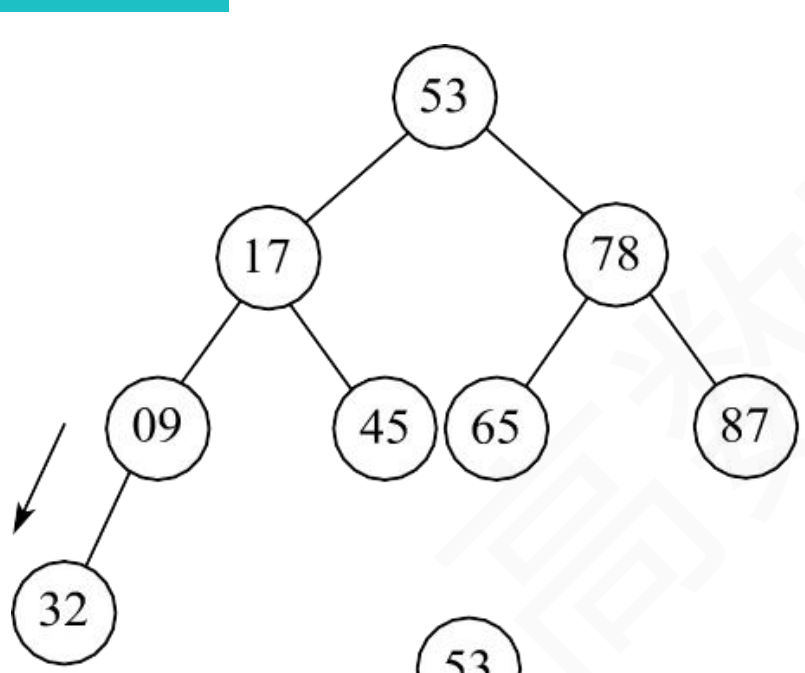
12.1 选择排序

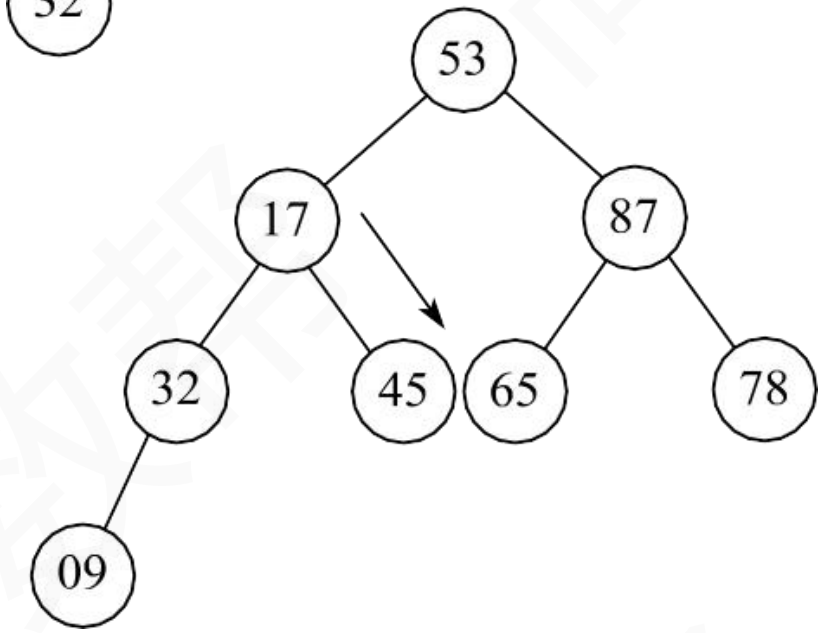
堆排序的关键是构造初始堆。

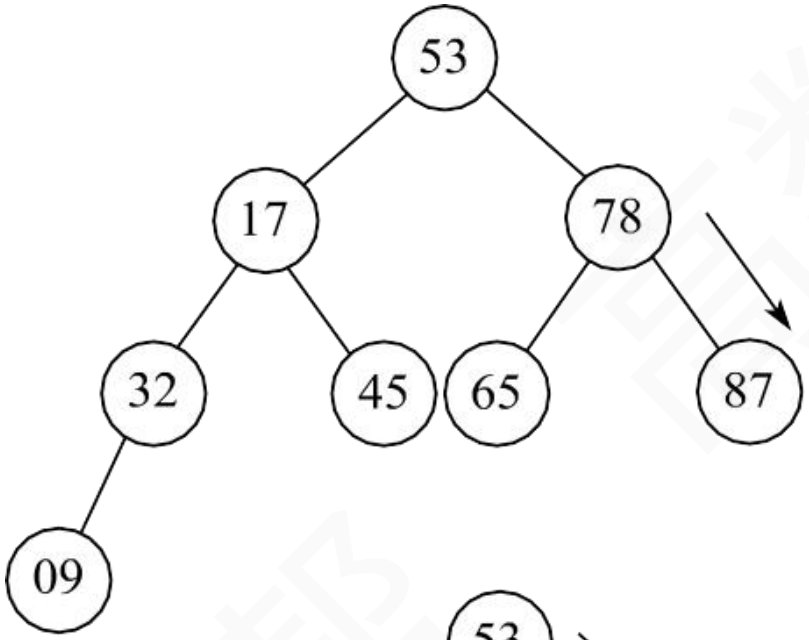
n个结点的完全二叉树，最后一个结点是第[n/2] 个结点的孩子。对第 [n/2] 个结点为根的子树进行筛选，使该子树成为堆。之后向前依次对各结点为根 的子树进行筛选，看该结点值是否大于其左右子结点的值，若不大于，则将左右子结点中 的较大值与之交换，交换后可能破坏下一级的堆，于是继续采用上述方法构造下一级的堆， 直到以该结点为根的子树构成堆为止。

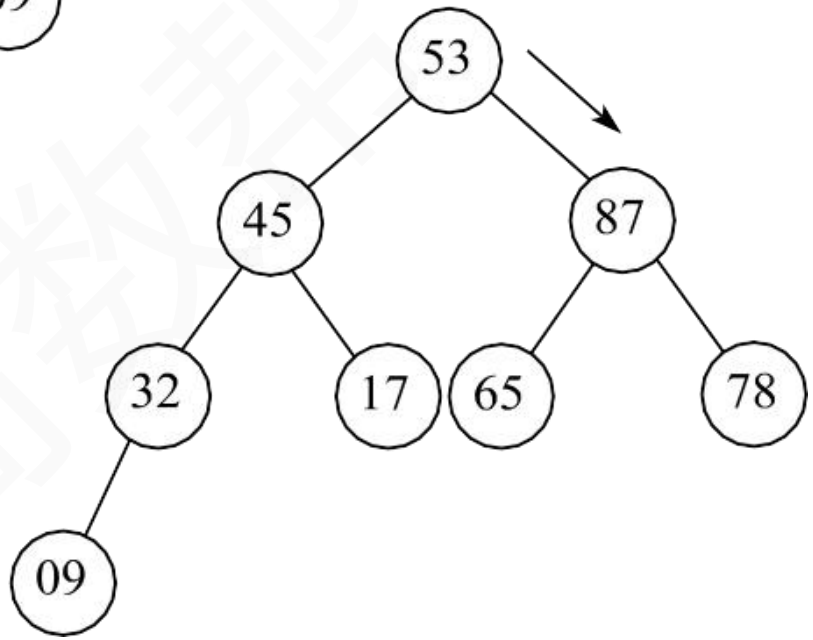
反复利用上述调整堆的方法建堆，直到根结点。

12.1 选择排序

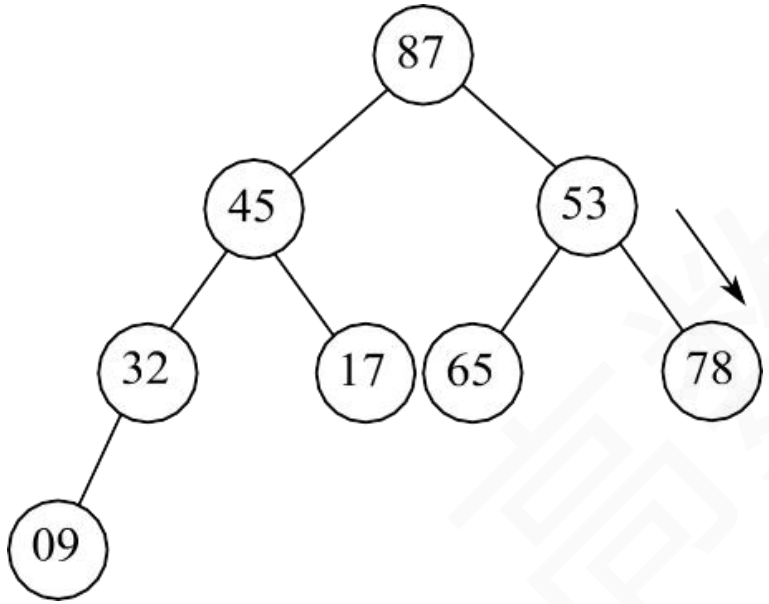

53
17
78
09
45
65
87
32

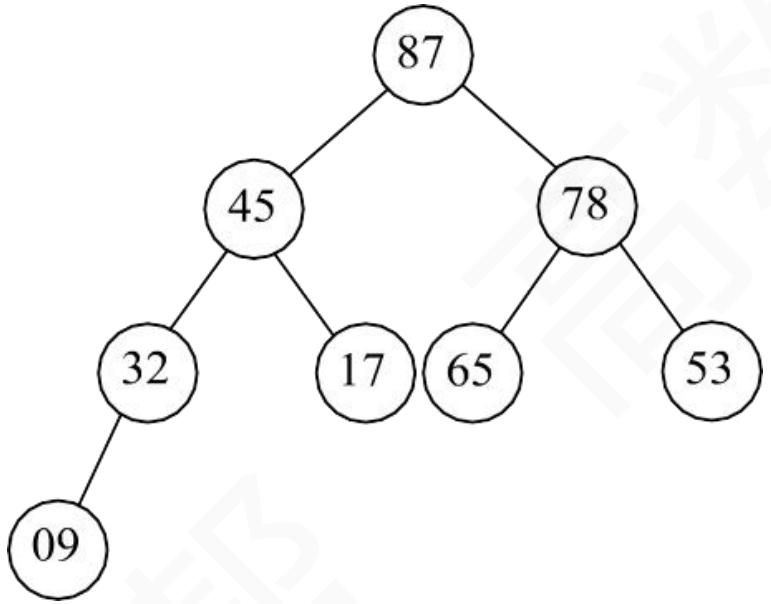

53
17
87
32
45
65
78
09


53
17
78
32
45
65
87
09


53
45
87
32
17
65
78
09

12.1 选择排序





题 1. 下面的序列中，（）是堆。

A. 1, 2, 8, 4, 3, 9, 10, 5

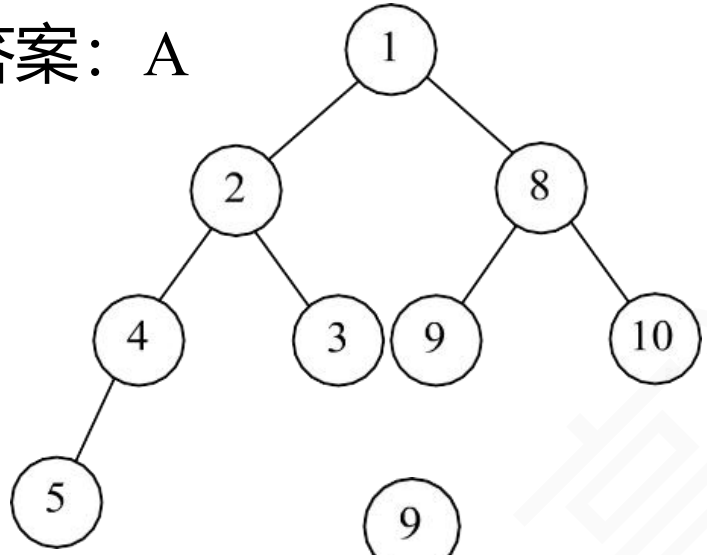
B. 1, 5, 10, 6, 7, 8, 9, 2

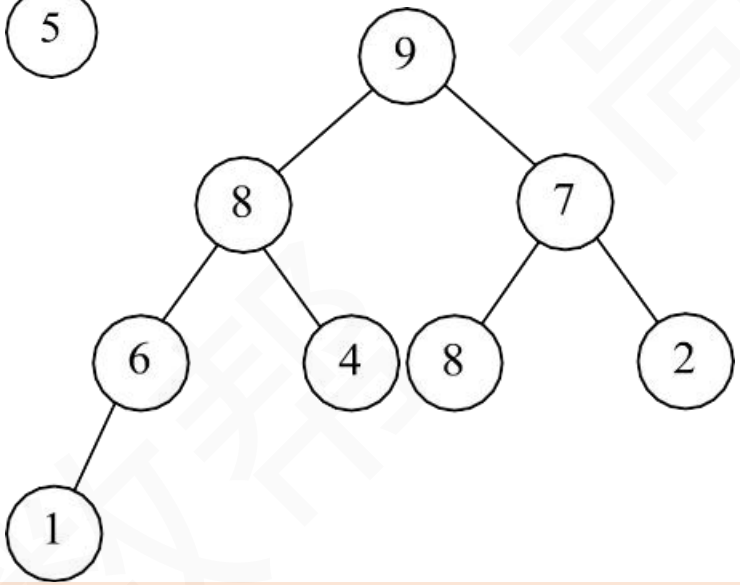
C. 9, 8, 7, 6, 4, 8, 2, 1

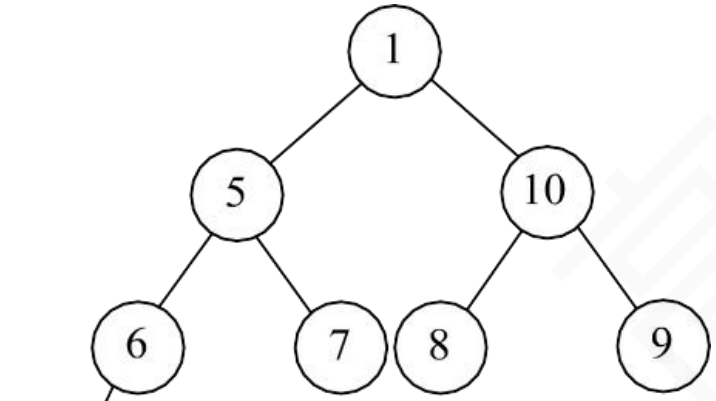
D. 9, 8, 7, 6, 5, 4, 3, 7

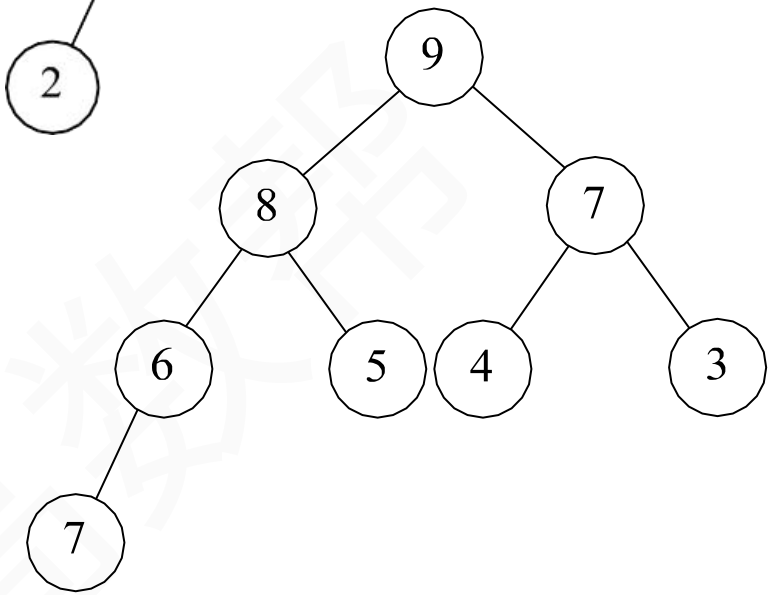
12.1 选择排序

答案：A



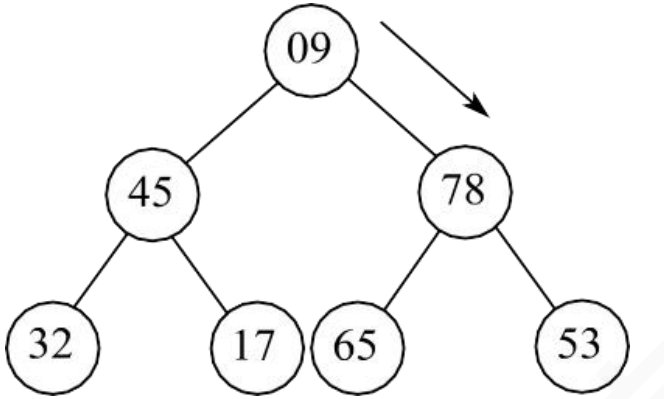


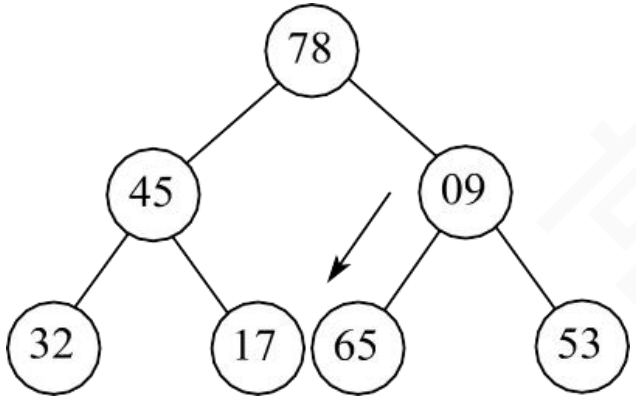


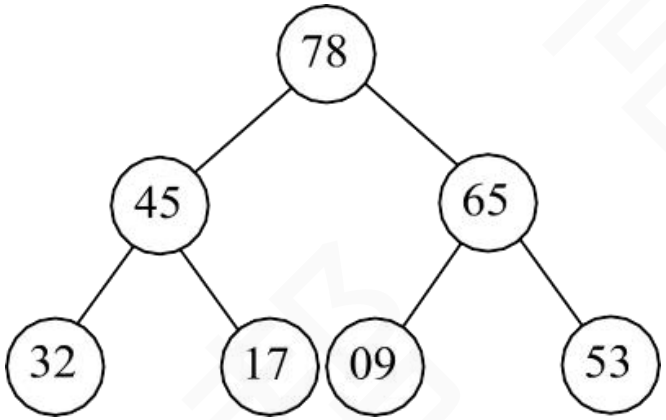


输出栈顶元素后，将堆的最后一个元素与栈顶元素交换，此时堆的性质被破坏。

12.1 选择排序

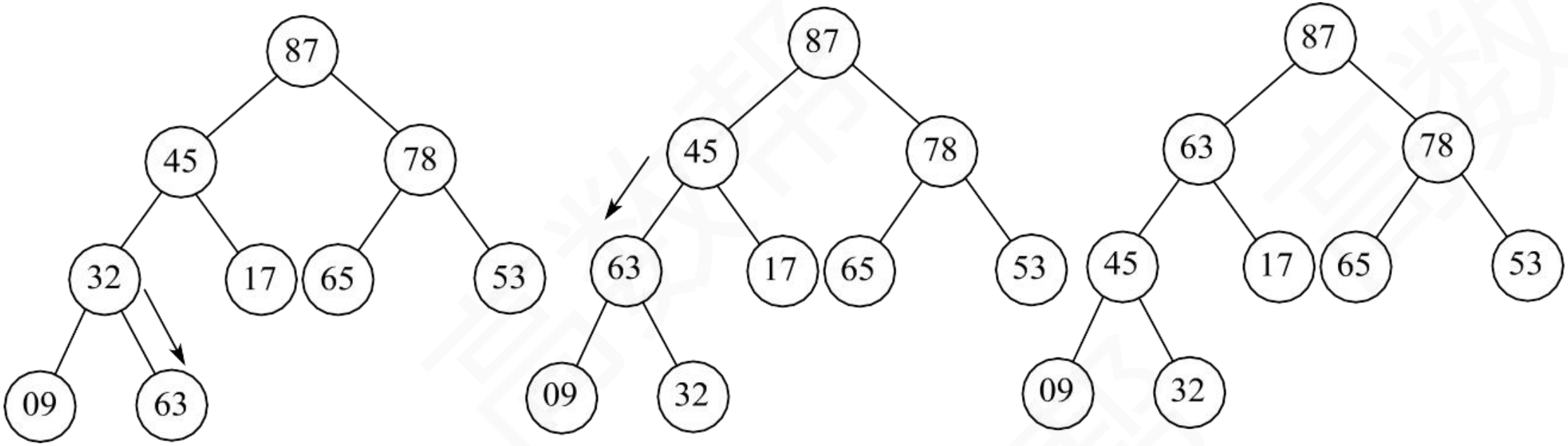






同时，堆也支持插入操作。对堆进行插入操作时，先将新结点放在堆的末端，再对这个新 结点向上执行调整操作。

12.1 选择排序



空间效率：$O(1)$

时间效率：$O(n\log_2 n)$

稳定性：不稳定。

例如，表L={1,2,2’} ，构造初始堆时可能将 交换到堆顶，最终排序序列也是L={2’,2,1,}。

题 2.设有 5000 个待排序的记录关键字，如果需要用最快的方法选出其中最小的 10 个记录关 键字，则用下列（ ）方法可以达到目的。

A.快速排序　　B.堆排序

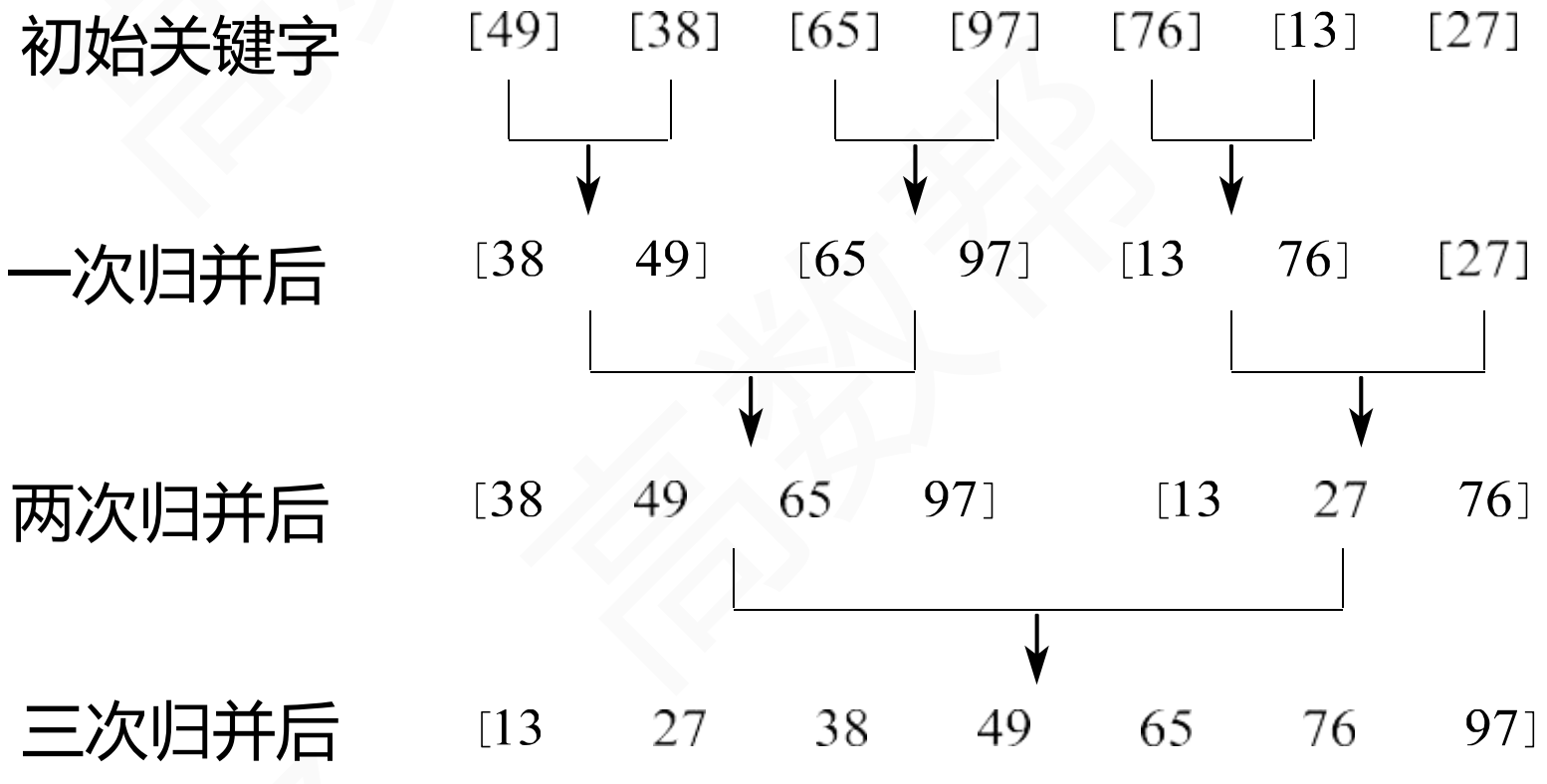
C.归并排序　　D.直接插入排序

答案：B

12.2 归并排序

“归并”的含义是将两个或两个以上的有序表组合成一个新的有序表。

假定待排序表含有n 个记录，则可将其视为n 个有序的子表，每个子表的长度为 1 ，然后两两归并，得到 [n/2] 个长度为 1 或 2 的有序表；继续两两归并……如此反复，直到合并 成一个长度为 n 的有序表为止，这种排序方法称为 2 路归并排序。



12.2 归并排序

“归并”的含义是将两个或两个以上的有序表组合成一个新的有序表。

假定待排序表含有n 个记录，则可将其视为n 个有序的子表，每个子表的长度为 1，然后两两归并，得到 [n/2] 个长度为 1 或 2 的有序表；继续两两归并……如此反复，直到合并 成一个长度为 n 的有序表为止，这种排序方法称为 2 路归并排序。

空间效率：$O(1)$

时间效率：$O(n\log_2 n)$

稳定性：稳定。

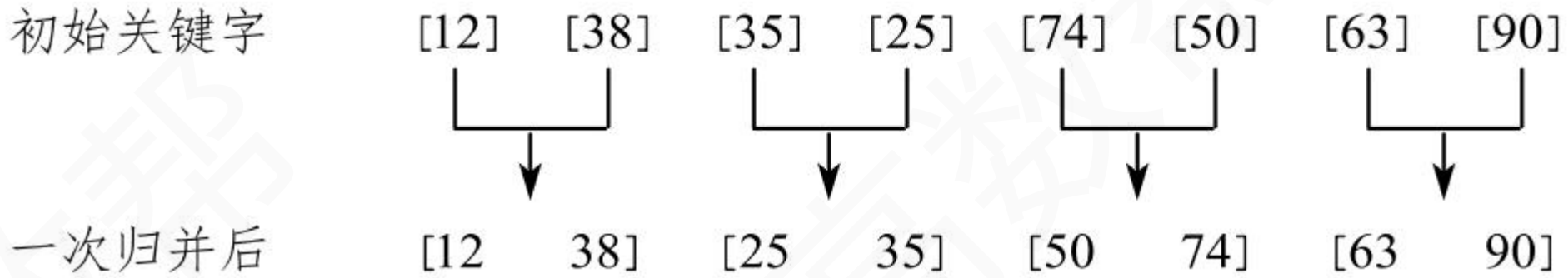
12.2 归并排序

题 1.一组记录值为(12, 38, 35, 25, 74, 50, 63, 90)，按 2 路归并排序方法对序列进行一趟归并后的结果是（　）。

A.12, 38, 25, 35, 50, 74, 63, 90　　B.12, 38, 35, 25, 74, 50, 63, 90

C.12, 25, 35, 38, 50, 74, 63, 90　　D.12, 35, 38, 25, 63, 50, 74, 90

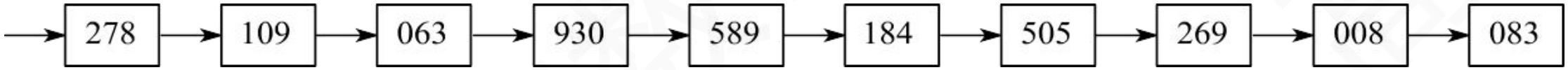
答案：A



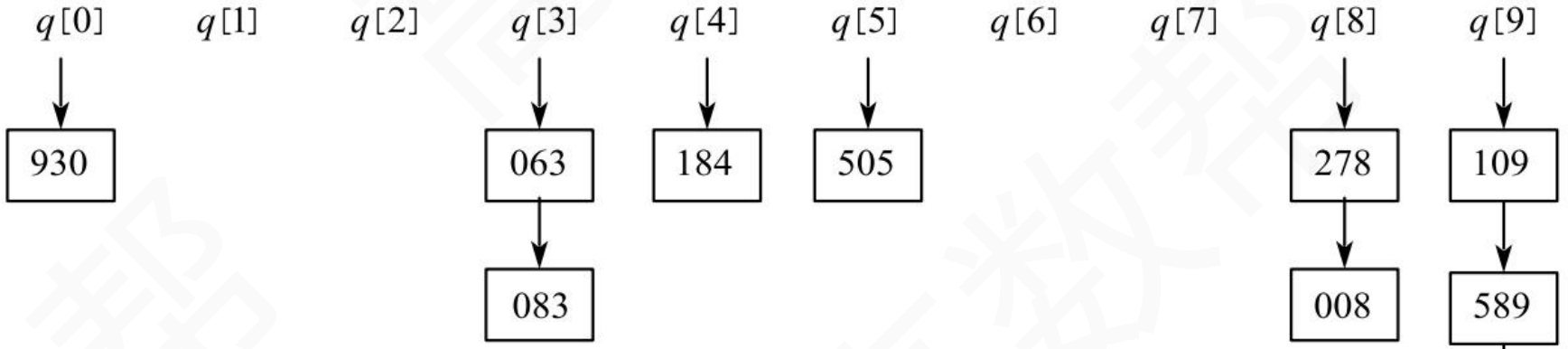
12.3 基数排序

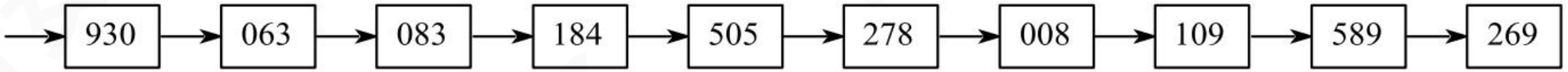
基数排序不基于比较和移动进行排序，而基于关键字各位的大小排序。

通常采用链式基数排序，假设对如下 10 个记录：



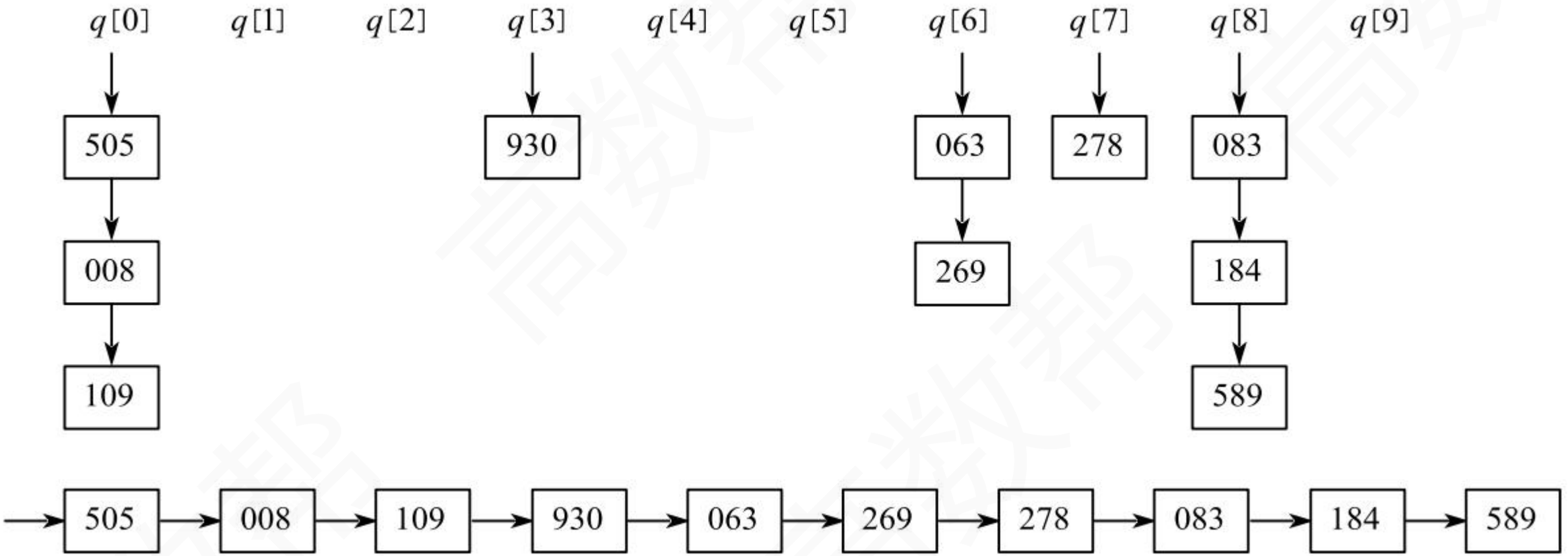
第一趟基于个位：





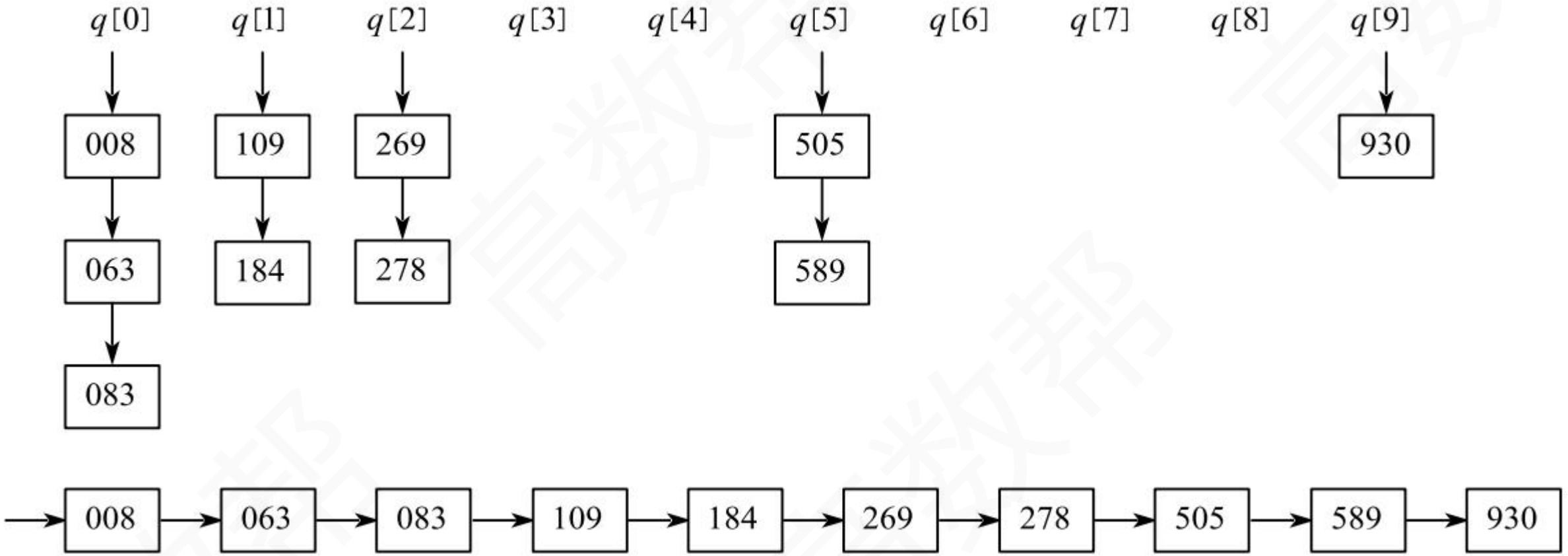
12.3 基数排序

第二趟基于十位：



12.3 基数排序

第三趟基于百位：



12.3 基数排序

空间效率：$O(r)$

时间效率：$O(d(n+r))$，它与序列的初始状态无关。

稳定性：稳定。

题 1.如果将所有中国人的按照生日（不考虑年份，只考虑月、日）来排序，那么使用下列排 序算法中的（ ）算法最快。

A.归并排序　　B.希尔排序

C.快速排序　　D.基数排序

答案：D

12.4 各种排序算法的比较

由于希尔排序的时间复杂度依赖于增量函数，所以无法准确给出其时间复杂度。

算法种类	时间复杂度			空间复杂度	是否稳定
	最好情况	平均情况	最坏情况		
直接插入排序	$O(n)$	$O(n^2)$	$O(n^2)$	$O(1)$	是
希尔排序				$O(1)$	否
冒泡排序	$O(n)$	$O(n^2)$	$O(n^2)$	$O(1)$	是
快速排序	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n^2)$	$O(\log_2 n)$	否
简单选择排序	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(1)$	否
堆排序	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(1)$	否
2路归并排序	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n)$	是
基数排序	$O(d(n+r))$	$O(d(n+r))$	$O(d(n+r))$	$O(r)$	是

12.4 各种排序算法的比较

题1.在插入和选择排序中，若初始数据基本正序，则选用_____，若初始数据基本反序，则选用_____。 答案：递增排列 递减排列

题2.在插入排序、希尔排序、选择排序、快速排序、堆排序、归并排序和基数排序中，排序是不稳定的有_____。 答案：选择排序、希尔排序、堆排序

题 3.以下四种排序方法中，需要附加的内存空间最大的是（ ）。 答案：D

A.插入排序 B.选择排序 C.快速排序 D.归并排序

题 4.一趟排序结束后不一定能够选出一个元素放在其在最终位置上的是（ ）。

A.简单选择排序 B.冒泡排序 C.快速排序 D.希尔排序

答案：D