10.1

◆10.1① 以关键码序列(503,087,512,061,908,170,897,275,653,426)为例,手工执行以下排序算法,写出每一趟排序结束时的关键码状态:

(1) 直接插入排序; (2) 希尔排序(增量 d[1]=5);

(3) 快速排序; (4) 堆排序;

(5) 归并排序; (6) 基数排序。

503 087 512 061 908 170 897 275 653 426

10

(1) 直接插入排序

第i趟	关键码状态
1	061 087 512 503 908 170 897 275 653 426
2	061 087 512 503 908 170 897 275 653 426
3	061 087 170 503 908 512 897 275 653 426
4	061 087 170 275 908 512 897 503 653 426
5	061 087 170 275 426 512 897 503 653 908
6	061 087 170 275 426 503 897 512 653 908
7	061 087 170 275 426 503 512 897 653 908
8	061 087 170 275 426 503 512 653 897 908
9	061 087 170 275 426 503 512 653 897 908
10	061 087 170 275 426 503 512 653 897 908

(2) 希尔排序 (增量 d[l]=5)

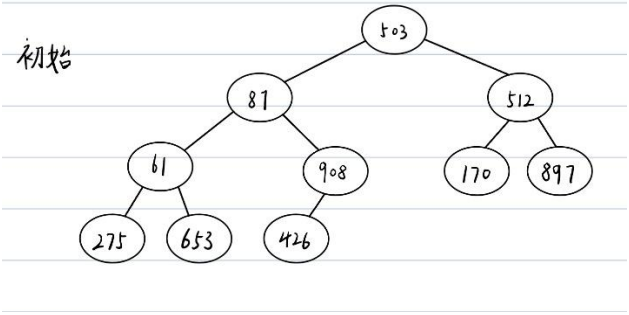
	增量	数组下标									
第i趟		0	1	2	3	4	5	6	7	8	9
1	5	170	087	512	061	908	503	897	275	653	426
2	5	170	087	512	061	908	503	897	275	653	426
3	5	170	087	275	061	908	503	897	512	653	426
4	5	170	087	275	061	908	503	897	512	653	426
5	5	170	087	275	061	426	503	897	512	653	908
6	2	170	087	275	061	426	503	653	512	897	908
7	2	170	061	275	087	426	503	653	512	897	908
8	1	061	087	170	275	426	503	512	653	897	908

(3) 快速排序

		数组下标									
第i趟	pivot	0	1	2	3	4	5	6	7	8	9
1	503	426	087	275	061	170	503	897	908	653	512
2	426	170	087	275	061	426					
3	170	061	087	170	275						
4	061	061	087	170							
5	087		087	170							
6	170			170							
7	897							512	653	897	908
8	512							512	653		
9	653								653		
10	908										908
11	完成	061	087	170	275	426	503	512	653	897	908

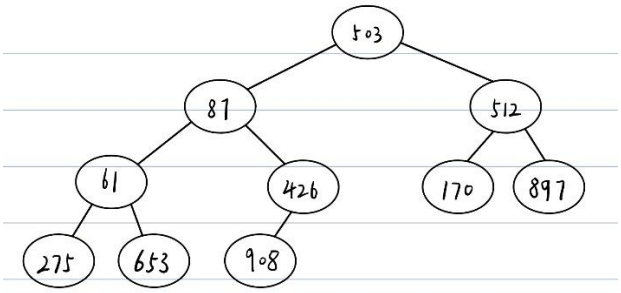
（4）堆排序

初始二叉树如图：n=10

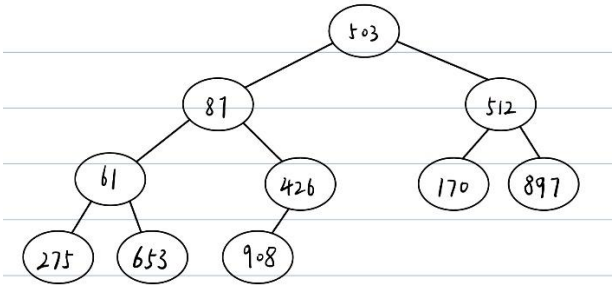


先自底向上将该序列调整为堆

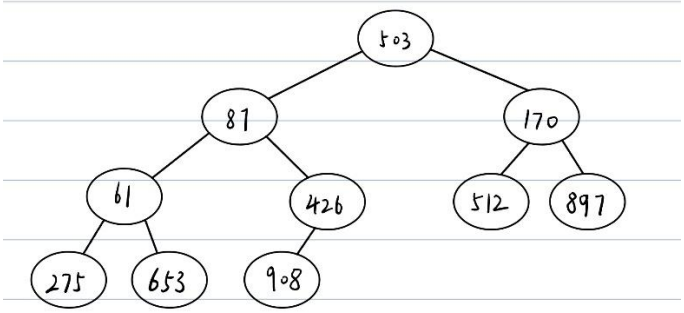
定位最后一个非叶子节点 i=(n-2)/2=4, a[4] = 908，从其开始向下调整得：



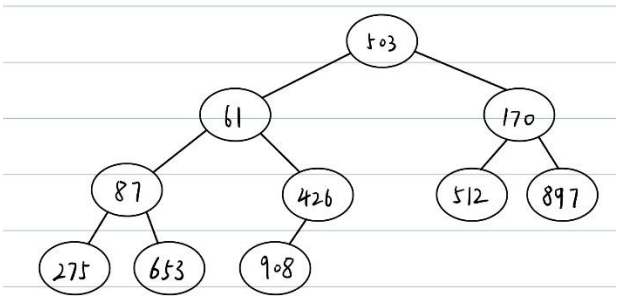
再从 a[3]=61 开始向下调整得：无需交换



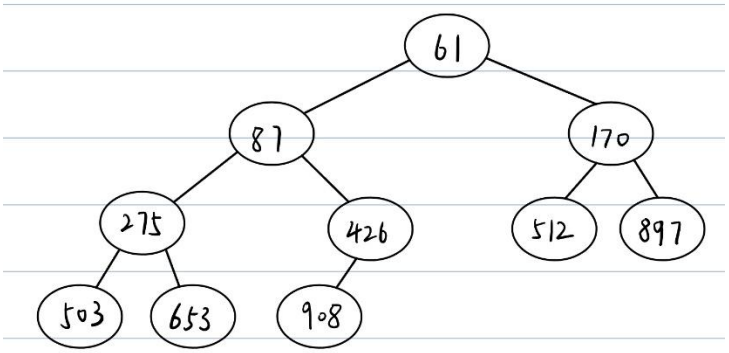
再从 a[2]=512 开始向下调整得：



再从 a[1]=87 开始向下调整得：

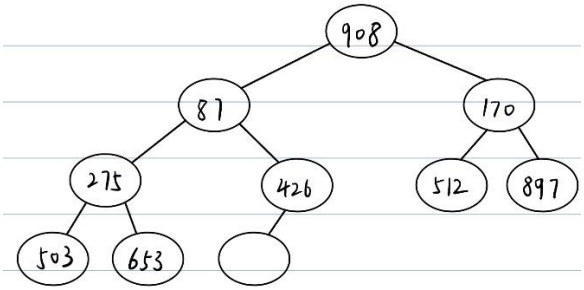


再从 a[0]=503 开始向下调整得：完成建堆操作

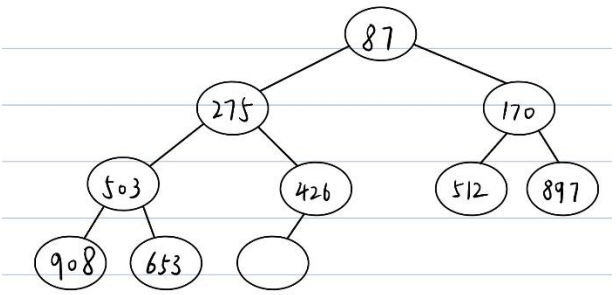


现在开始输出堆顶并调整

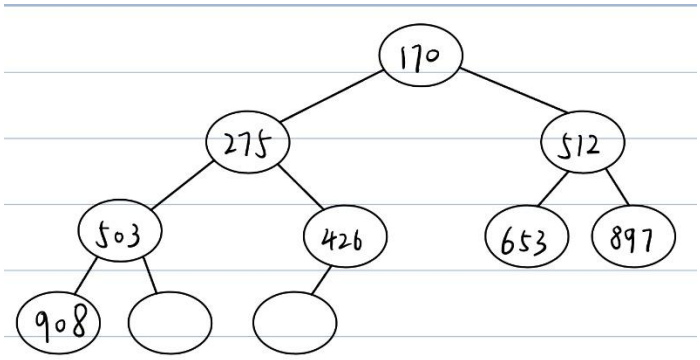
输出 61，交换堆顶和 908：



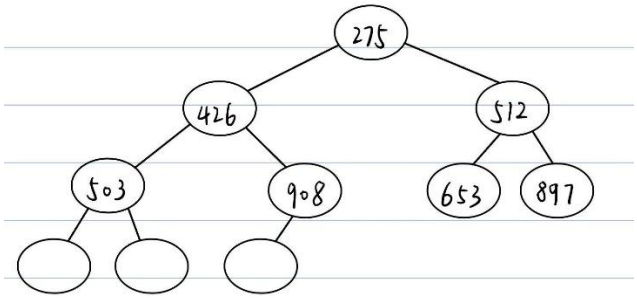
调整后得:



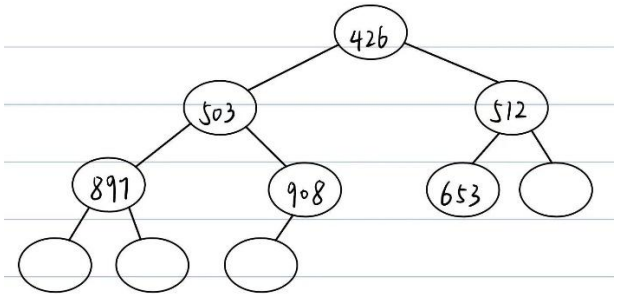
输出 87，交换堆顶和 653，调整后得：



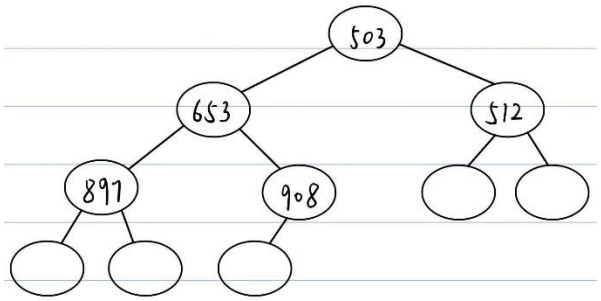
输出 170，交换堆顶和 908，调整后得：



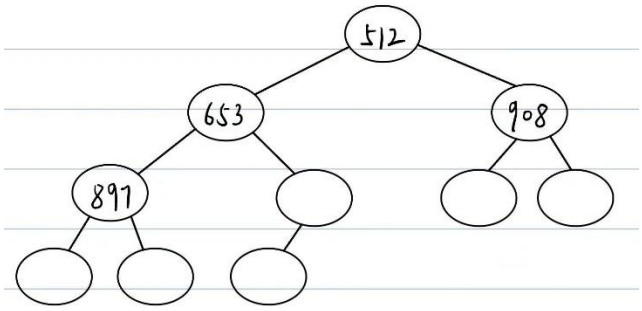
输出 275，交换堆顶和 897，调整后得：



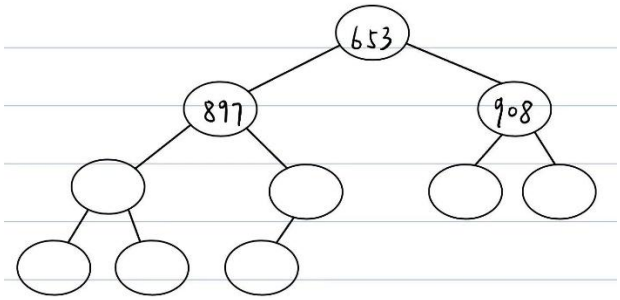
输出 426，交换堆顶和 653，调整后得：



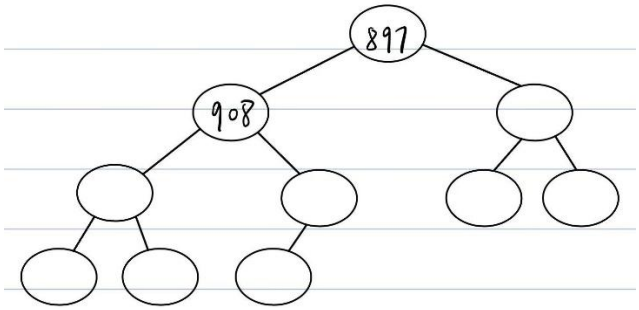
输出 503，交换堆顶和 908，调整后得：



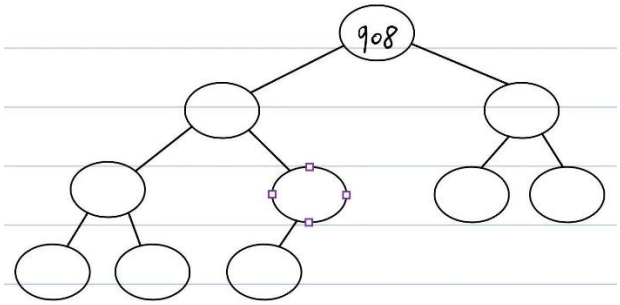
输出 512，交换堆顶和 897，调整后得：



输出 653，交换堆顶和 908，调整后得：



输出 897，交换堆顶和 908，调整后得：



输出 908

最终输出序列为 061 087 170 275 426 503 512 653 897 908

（5）归并排序

第i趟	数组下标	0	1	2	3	4	5	6	7	8	9
0	有序段序号	0	1		2		3		4		5
	有序段	503	087	512	061	908	170	897	275	653	426
1	有序段序号	0			2		3		4		5
	有序段	087	503	512	061	908	170	897	275	653	426
2	有序段序号	0			2		3		4		
	有序段	087	503	512	061	908	170	897	275	426	653
3	有序段序号	0					3		4		
	有序段	061	087	503	512	908	170	897	275	426	653
4	有序段序号	0					3				
	有序段	061	087	503	512	908	170	275	426	653	897
5	有序段序号	0									
	有序段	061	087	170	275	426	503	512	653	897	908

（6）基数排序

第i趟	操作	f[j]										
1	分配	0	170									
		1	061									
		2	512									
		3	503	653								
		4										
		5	275									
		6	426									
		7	087	897								
		8	908									
		9										
	回收	array	170	061	512	503	653	275	426	087	897	908
2	分配	0	503	908								
		1	512									
		2	426									
		3										
		4										
		5	653									
		6	061									
		7	170	275								
		8	087									
		9	897									
	回收	array	503	908	512	426	653	061	170	275	087	897
3	分配	0	061	087								
		1	170									
		2	275									
		3										
		4	426									
		5	503	512								
		6	653									
		7										
		8	897									
		9	908									
	回收	array	061	087	170	275	426	503	512	653	897	908

10.3

10.3② 试问在 10.1 题所列各种排序方法中，哪些是稳定的？哪些是不稳定的？并为每一种不稳定的排序方法举出一个不稳定的实例。

（1）稳定的：直接插入排序、归并排序、基数排序

不稳定的：希尔排序、快速排序、堆排序

（2）不稳定实例

希尔排序（假设初始增量为 2)：2 2 1 8 9 5 ，其中第一个 2 会被调整到第二个 2 的后面。

快速排序（以第一个元素为标杆)：3 9 2 5 2，第二个 2 会被放到 3 的位置上，即放在了第一个 2 的前面。

堆排序：调整堆时，子节点的 key 值与父节点相等时，子节点会上升，如

原始数组: (5, A) (3, B) (5, C) (2, D) (3, E)

堆排序后: (2, D) (3, E) (3, B) (5, C) (5, A)

10.15

◆10.15④ 对一个由 n 个关键字不同的记录构成的序列，你能否用比 $2n-3$ 少的次数选出这 n 个记录中关键字取最大值和关键字取最小值的记录？若能，请说明如何实现？在最坏情况下至少进行多少次比较？

设序列为 a1, a2, ..., an

设置两个变量 max 和 min，分别用来记录最大值和最小值。设置工作指针 i。

1. 初始化 max，min：

若 n 为奇数，令 max=min=a1，i=2；

若 n 为偶数，比较 a1 和 a2，max=较大者，min=较小者，i=3

2. 成对处理剩余元素：

每次取 ai 和 ai+1：

比较两者；

将两者中较大者与 max 比较，若比 max 大，则替换 max；

将两者中较小者与 min 比较，若比 min 小，则替换 min。

直到处理完所有元素。

（处理每对数据时需要 3 次比较）

设 n=2k 或 2k+1，

k≥2 时，共需要 3k = [n]×3/2 次比较。由于 n≥4，有[n]×3/2 ≤ 3n/2 < 2n-3

k=1 时，比较次数等于 2n-3。

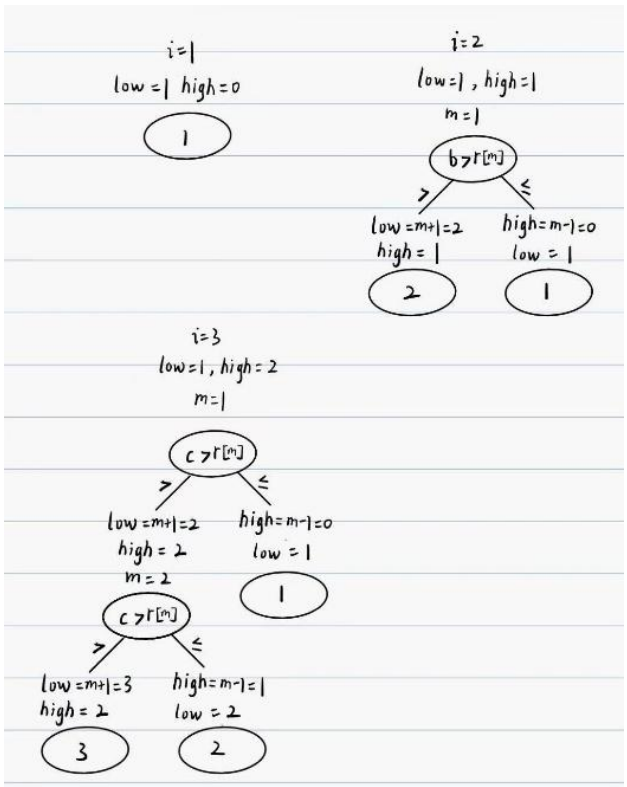
10.21

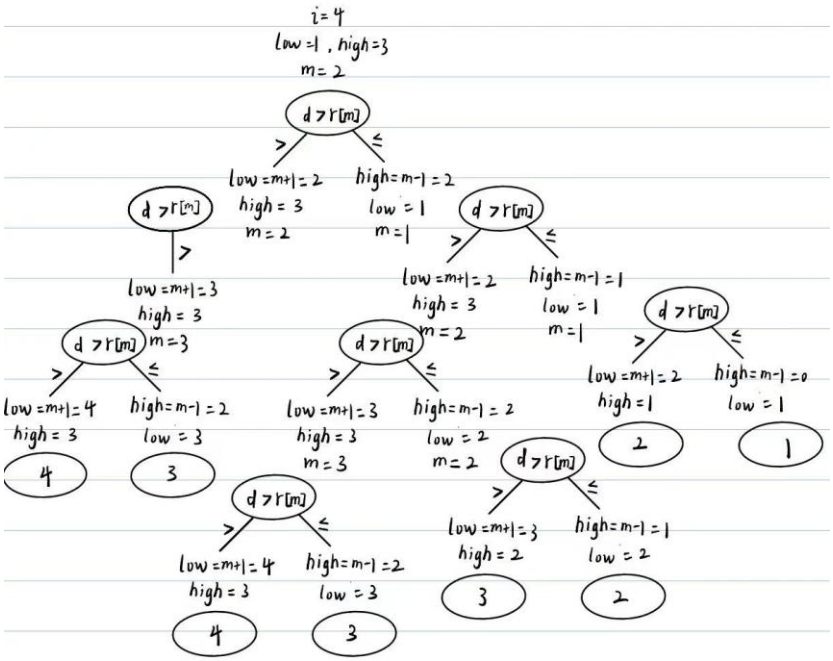
10. 21③ 分别利用折半插入排序法和 2-路归并排序法对含 4 个记录的序列进行排序，画出描述该排序过程的判定树，并比较它们所需进行的关键字间的比较次数的最大值。

设记录序列为[a, b, c, d]

折半插入排序：

r[0]存储当前插入元素，r[1:4]=[a, b, c, d]

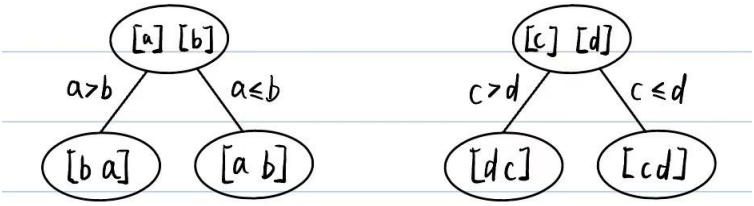




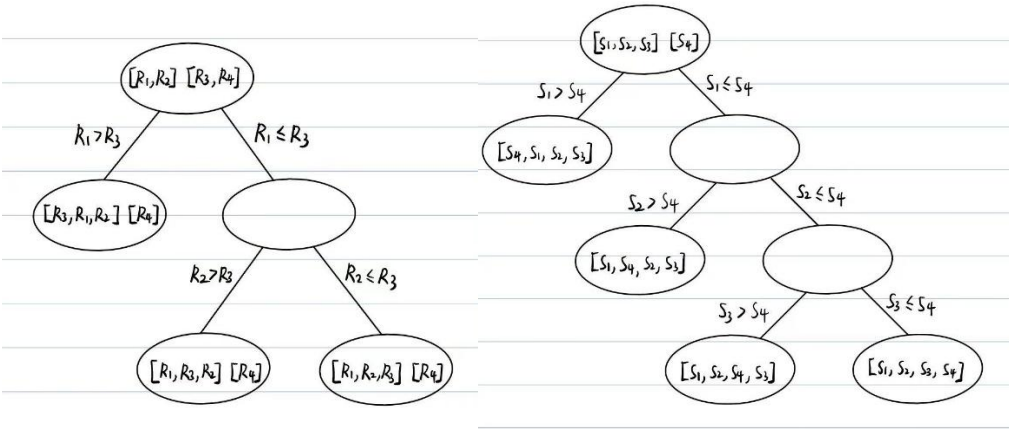
最多比较 0+1+2+4=7 次

2-路归并排序：

对两组最小子序列做归并，得到两个子序列[R1, R2][R3, R4]：



对[R1, R2][R3, R4]做归并：



最多比较 1+1+2+3=7 次。故两种算法的最多比较次数相同。

(Optional)

11.1

◆11.1① 假设某文件经内部排序得到 100 个初始归并段，试问：

（1）若要使多路归并三趟完成排序，则应取归并的路数至少为多少？

（2）假若操作系统要求一个程序同时可用的输入、输出文件的总数不超过 13，则按多路归并至少需几趟可完成排序？如果限定这个趟数？则可取的最低路数是多少？

（1）由 $\log_k 100 \leq 3$ 得 $k \geq e^{(1n(100) / 3)} \approx 4.6$

则归并路数至少为 5。

（2）一次 k 路归并操作需要 k 个输入文件和 1 个输出文件，要求 k+1≤13，则 k≤12。设趟数为 p，要求 $k^p \geq 100$。

若取 p=1，显然 k 不满足约束。若取 p=2，则 10≤k≤12

因此，至少需要 2 趟完成排序。如果限定 2 趟，可取的最低路数为 10 路。

11.2

◆11.2② 假设一次 I/O 的物理块大小为 150，每次可对 750 个记录进行内部排序，那么，对含有 150000 个记录的磁盘文件进行 4-路平衡归并排序时，需进行多少次 I/O？

归并初始段数 s=150000/750=200 个

设趟数为 p，$4^p \geq 200$，则求得 $p \geq 3.8219$，即 p 最小取 4。

生成初始归并段的 I/O 次数为 150000/150*2=2000 次

每趟归并排序的 I/O 次数也为 150000/150*2=2000 次

则总 I/O 次数=2000+2000*4=10000 次

11.5

11.5② 为什么置换-选择排序能得到平均长度为 2w 的初始归并段？能否依置换-插入或置换-交换等策略建立类似的排序方法？

（1）工作区的初始容量长度为ω。设工作区的最小记录为 r，假设输入序列为随机分布时，每次读入的新纪录有 0.5 的概率≥r，从而能被划分到该归并段，且对工作区初始的每个记录以及后来加入该归并段的记录都是如此，即有该归并段长度 L = ω+0.5L，求得 L=2ω。

（2）不可以。插入/交换策略均需维持工作区全序，会使工作区丧失筛选不同归并段记录数据的能力。

11.11

◆11.11② 已知某文件经过置换选择排序之后，得到长度分别为 47，9，39，18，4，12，23 和 7 的八个初始归并段。试为 3 路平衡归并设计一个读写外存次数最少的归并方案，并求出读写外存的次数。

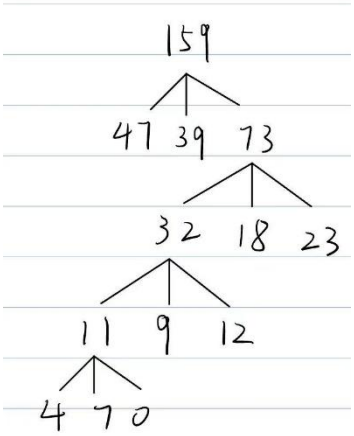
构造 3 路哈夫曼树：

设 3 路哈夫曼树中度为 0 的结点个数为 n_0，度为 3 的结点个数为 n_3，则有 $n_0+n_3=3n_3+1$。

则 $n_0=2n_3+1$，即叶子数 n_0 应满足 $n_0=2k+1$。

由于只有 8 个初始归并段，所以需要添加一个长度为 0 的虚段。

构造出的 3 路哈夫曼树如图：



第一趟读入 4、7、0 三个段，写入 11 这个新段；

第二趟读入 11、9、12 三个段，写入 32 这个新段；

第三趟读入 32、18、23 三个段，写入 73 这个新段；

第三趟读入 47、39、73 三个段，写入 159 这个新段；

访问外存次数 = (11+32+73+159)*2 = 550