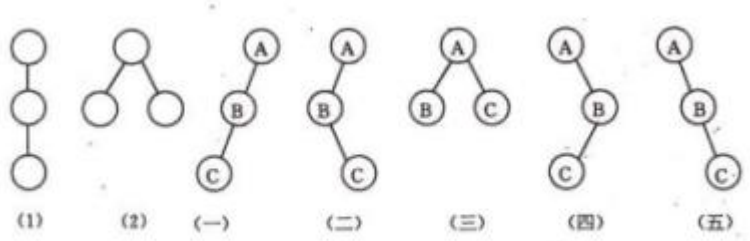
6.3 试分别画出具有 3 个结点的树和 3 个结点的二叉树的所有不同形态。

含三个结点的树只有两种形态：（1）和（2）；

而含 3 个结点的二叉树可能有下列 5 种形态



注意，(2)和(三)是完全不同的结构。

6.6 已知在一棵含有 n 个结点的树中，只有度为 k 的分支结点和度为 0 的叶子结点。试求该树含有的叶子节点数目。

解：利用上题结论易得结果。设度为 k 的结点个数为 n_k ，则总结点数为 $n=1+n_k k$ 。叶子结点的数目应等于总结点数减去度不为 0 的结点的数目，即

$$n_0 = n - n_k = n - \frac{n-1}{k}$$

6.7 一棵含有 n 个结点的 k 叉树，可能达到的最大深度和最小深度各为多少？

注：此题求最小深度时，部分没有计算具体值，只写了深度最小为完全 K 叉树；对其结果取整混乱，向上取向下取的都有；注意取值符号的书写。

解：能达到最大深度的树是单支树，其深度为 n。满 k 叉树的深度最小，其深度为

$$\lceil \log_k (n(k-1)+1) \rceil$$

6.10 对于那些所有非叶子结点均含有左右子数的二叉树：

(1) 试问：有 n 个叶子结点的树中共有多少个结点？

(2) 试证明：$\sum_{i=1}^{n} 2^{-(l_i-1)} = 1$ ，其中 n 为叶子结点的个数，l_i 表示第 i 个叶子结点所在的层次（设根节点所在层次为 1）。

注：此题（1）正确率很高，（2）1/3 同学证明极其简略，包括少数没做。

解：（1）总结点数为$1+2n_1$，其中n_1为非叶子结点数，则叶子结点数为$n=n_1+1$，所以总结点数为$2n-1$。

（2）用归纳法证明。

i=1，说明二叉树只有一个叶子结点，则整棵树只有一个根结点，$l_1=1$，

$2^{-(l_1-1)}=1$，结论成立。

设有 n 个叶子结点时也成立，即$2^{-(l_1-1)}+2^{-(l_2-1)}+\ldots+2^{-(l_p-1)}+\ldots+2^{-(l_n+1)}=1$，现假设增加一个叶子结点，这意味着在某叶子结点 p 上新生两个叶子结点，而结点 p 则成为非叶子结点，可见，总结点数增 2，叶子结点数增 1。此时，所有叶子结点是原结点除去 p，然后加上两个深度为l_p+1的新叶子结点，由此，

$$2^{-(l_1-1)}+2^{-(l_2-1)}+\ldots+2^{-(l_{p-1}-1)}+2^{-(l_{p+1}-1)}+\ldots+2^{-(l_n+1)}+2^{-(l_p+1-1)}+2^{-(l_p+1-1)}$$

$$=2^{-(l_1-1)}+2^{-(l_2-1)}+\ldots+2^{-(l_p-1)}+\ldots+2^{-(l_n+1)}=1$$

则当 i=n+1 时，也成立，由此即得到证明。

6.14 找出所有满足下列条件的二叉树：

(a) 它们在先序遍历和中序遍历时，得到的节点访问序列相同；

(b) 它们在后序遍历和中序遍历时，得到的结点访问序列相同；

(c) 它们在先序遍历和后序遍历时，得到的节点访问序列相同。

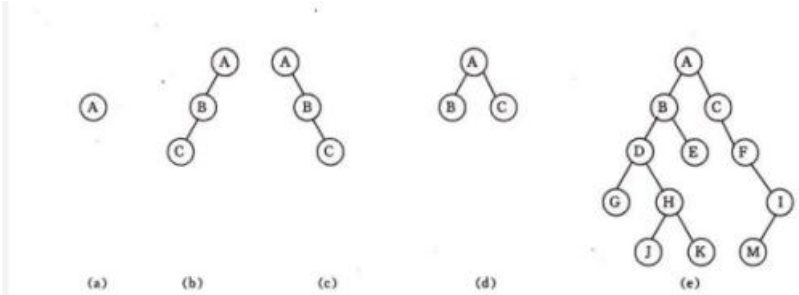
注：对此题理解较好，但是不够全面，少考虑了空树和只有一个结点的情况。

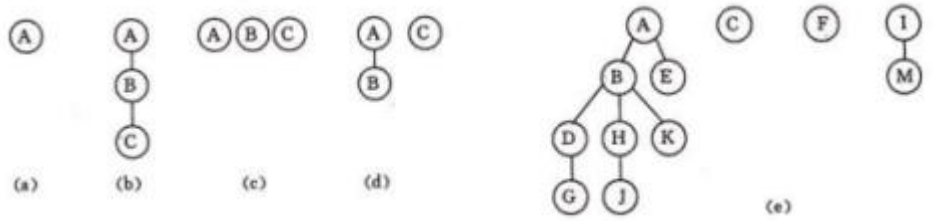
解：(a) 不含左子树的二叉树。

(b) 不含右子树的二叉树。

(c) 即不含左子树，也不含右子树的二叉树。

6.21 画出和下列二叉树相应的森林





6.30

证明：树中结点 u 是结点 v 的祖先，当且仅当在先序序列中 u 在 v 之前，且在后序序列中 u 在 v 之后。

注：此题

证明：

树的先序遍历算法：先访问树的根结点，然后依次先序遍历根的每棵子树。

树的后序遍历算法：先依次后序遍历根的每棵子树，然后访问树的根结点。

充分性：若要保证 u 在 v 之前。必须保证根节点的访问先于子树 即采用先序遍历，相反，要保证 u 在 v 之后，必须保证根结点的访问晚于子树，即采用后序遍历。

必要性：若采用先序遍历，必然先访问根结点，后访问其子树，固 u 的访问必先于 v，若采用后序序列，则根结点在最后被访问，因而 u 的访问必 v 之后。

故命题得证。

要点：

1、说明树的先序遍历算法、后序遍历算法。

2、论证树的先序遍历序列中 u 在 v 之前。

3、论证树的后序遍历序列中 u 在 v 之后。

7.1 已知如下图所示的有向图，请给出该图的

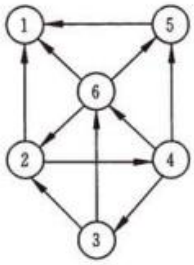
(1)每个顶点的入/出度；

(2)邻接矩阵；

(3)邻接表；

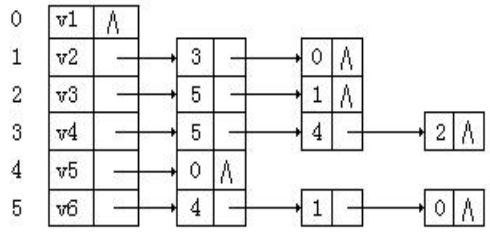
(4)逆邻接表；

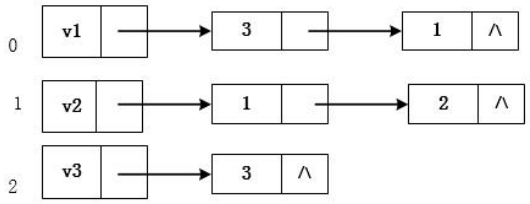
(5)强连通分量。



注：此题做的较好，问题主要出现在（3）和（4）的画法上，需注意。

错误示例：





解： (1) ID(1)=3 OD(1)=0

ID(2)=2 OD(2)=2

ID(3)=1 OD(3)=2

ID(4)=1 OD(4)=3

ID(5)=2 OD(5)=1

ID(6)=2 OD(6)=3

(2) 0 0 0 0 0 0

1 0 0 1 0 0

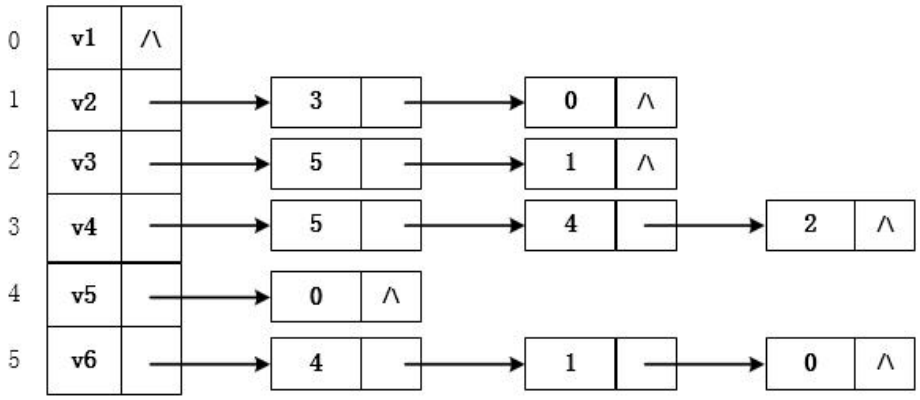
0 1 0 0 0 1

0 0 1 0 1 1

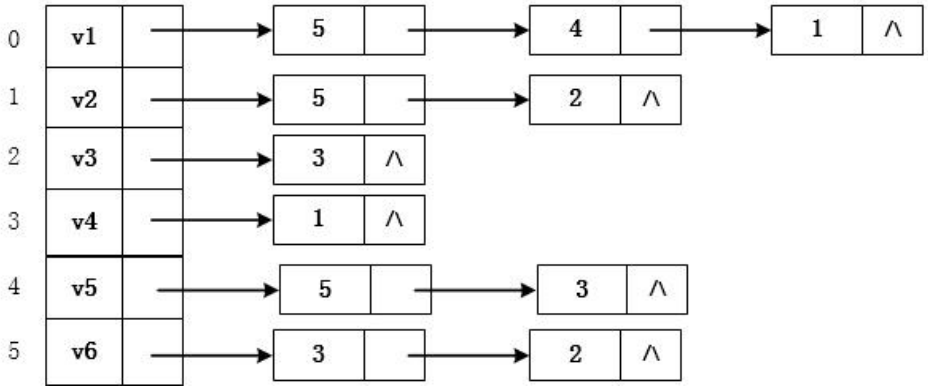
1 0 0 0 0 0

1 1 0 0 1 0

（3）



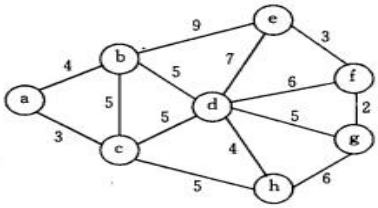
（4）



7.7 请对下图的无向带权图，

(1)写出它的邻接矩阵，并按普里姆算法求其最小生成树；

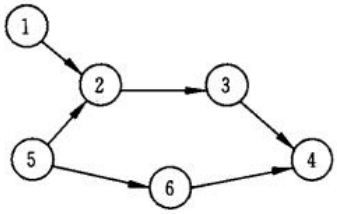
(2)写出它的邻接表，并按克鲁斯卡尔算法求其最小生成树。



题 7.7 图

注：同邻接表的画法。

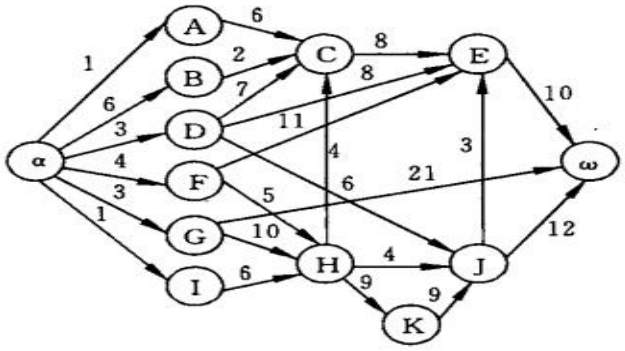
7.9 试列出下图中全部可能的拓扑有序序列，并指出应用 7.5.1 节中算法 Topological Sort 求得的是哪一个序列（注意：应先确定其存储结构）。



题 7.9 图

5	6	1			2			3			4
5		1	6		2			3			4
5		1			2	6		3			4
5		1			2			3	6		4
		1	5	6	2			3			4
		1	5		2	6		3			4
		1	5		2			3	6		4

7.10 对于下图所示的 AOE 网络，计算各活动弧的 e(ai)和 l(aj)函数值、各事件(顶点)的 ve(vi)和 vl(vj)函数值；列出各条关键路径。



题 7.10 图

注：1/6 的同学 ve(vi)和 vl(vj)值计算错误，关键路径没有写。

顶点	ve	vi
α	0	0
A	1	20
B	6	24
C	17	26
D	3	19
E	34	34
F	4	8
G	3	3
H	13	13
I	1	7
J	31	31
K	22	22
ω	44	44

边	e	j	j—e
(α,A)	0	19	19
(α,B)	0	18	18
(α,D)	0	16	16
(α,F)	0	4	4
(α,G)	0	0	0
(α,I)	0	6	6
(A,C)	1	20	19
(B,C)	6	24	18
(D,C)	3	19	16
(D, E)	3	26	23
(D,J)	3	25	22
(F,E)	4	23	19
(F, H)	4	8	4
(G,ω)	3	23	20
(G,H)	3	3	0
(I, H)	1	7	6
(C, E)	17	26	9
(H, C)	13	22	9
(H,J)	13	27	14
(H,K)	13	13	0
(K,J)	22	22	0
(J,E)	31	31	0
(J,ω)	31	32	1
(E,ω)	34	34	0

关键路径只有一条：（a,G,H,K,J,E,w）

7.37 试设计一个求有向无环图中最长路径的算法，并估计其时间复杂度。

注：1/4 的同学没有做，做了的思路都基本正确。

思路： Bellman-Ford 算法求最长路径，只要把图中的边权改为原来的相反数即可。也可以用 Floyd-Warshall 算法求每对节点之间的最长路经，因为最长路径也满足最优子结构性质，而 Floyd 算法的实质就是动态规划。但是，如果图中含有回路，Floyd 算法并不能判断出其中含有回路，且会求出一个错误的解；而 Bellman-Ford 算法则可以判断出图中是否含有回路。

9.2

9.15 试证明：高度为 h 的 2-3 树中叶子结点的数目在 2h-1 与 3h-1 之间。

在 2-3 树中，每个内部结点（非叶子结点）有两个或三个孩子，而且所有叶子都在同一层上；

一方面，若某棵 2-3 树 2 只包含 2-结点，则就是一棵满二叉树。高度为 h 的满二叉树的叶子结点数是 2^(h-1);

另一方面，若某棵 2-3 树只包含 3-结点，那么第 i 层结点数目是 3^(i-1)。高度为 h 的“3 树”的叶子结点数是 3^(h-1);

高度为 h 的 2-3 树的叶子结点数目在 2^(h-1)与 3^(h-1)之间。

9.16 在含有 n 个关键码的 m 阶 B-树中进行查找时，最多访问多少个结点？

注：同取整问题，可参考教科书 P241 证明。

(1)第一层为根，至少一个结点，跟至少有两个孩子，因此在第二层至少有两个结点;

(2)除根和树叶之外，其他结点至少有[m/2]个孩子，一次第三层至少有 2*[m/2]个结点，在第四层至少有 2*[m/2]^2 个结点;

(3)那么在第 h+1 层至少有 2*[m/2]^(h-1)个结点，而 h 层的结点个数 n+1，有：

n+1>2*⌈m/2⌉^(h-1)

$h<=\log_{\lceil m/2 \rceil}\{(n+1)/2\}+1$

9.21 在地址空间为 0~16 的散列区中　对以下关键字序列构造两哈希表　。

Jan,Feb,Mar,Apr,May,June,July,Aug,Sep,Oct,Nov,Dec

(1)用线性探测开放定址法处理冲突

(2)用链地址法处理。

并分别求这两个哈希表在等概率情况下查找成功和不成功时的平均查找长度。

注：求查找不成功时的平均查找长度时错误率高（不知道分母值是多少）；

哈希表画的不完整，注意表长。

解：设哈希函数为 H(x)=i/2 取整,其中 i 为关键字中第一个字母在字母表中的序号。

因为：

H(Jan)=5;

H(Feb)=3;

H(Mar)=6;

H(Apr)=0;

H(May)=6; H1(May)=7;

H(June)=5;H1(June)=6;H2(June)=7;H3(June)=8

H(July)=5;H1(July)=6;H2(July)=7;H3(July)=8;H4(July)=9;

H(Aug)=0; H1(Aug)=1

H(Sep)=9; H1(Sep)=10

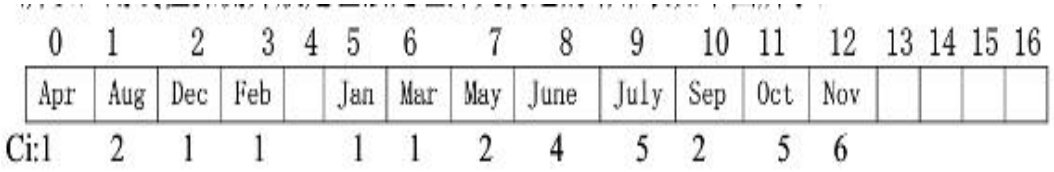
H(Oct)=7; H1(Oct)=8;H2(Oct)=9; H3(Oct)=10; H4(Oct)=11;

H(Nov)=7;H1(Nov)=8;H2(Nov)=9;H3(Nov)=10;

H4(Nov)=11;H5(Nov)=12;

H(Dec)=2

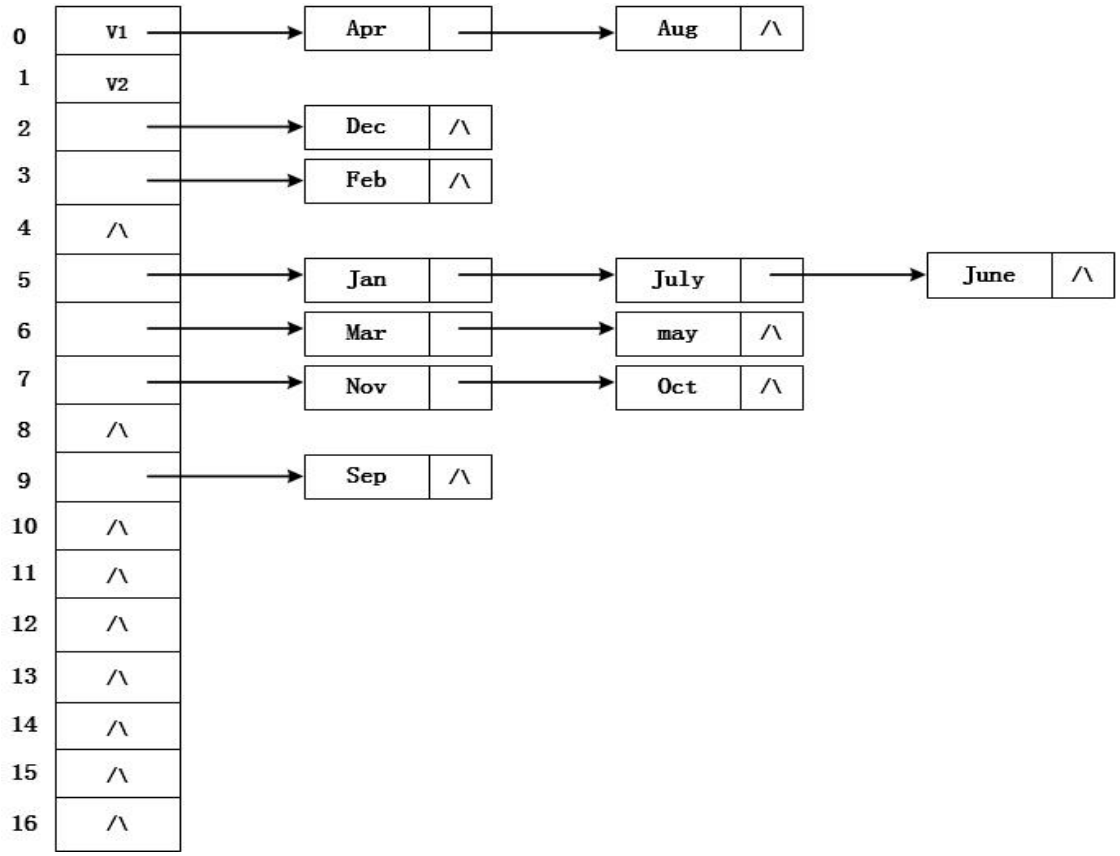
所以　用线性探测开放定址法处理冲突构造的哈希表如下图所示



并求得　ASL 成功=（1*5+2*3+4+5*2+6）/12=31/12

ASL 不成功=（5+4+3+2+1+9+8+7+6+5+4+3+2+1）/14=60/14

（2）用链地址法处理冲突构造的哈希表如下图所示：



ASL 成功=（1*7+2*4+3）/12=18/12

ASL 不成功=（1*3+2*3+3）/14=12/14

9.38 试写一算法，将两棵二叉排序树合并为一棵二叉排序树。

注：做的大多雷同，部分没写。

算法思路：

假定将以 h 为头指针的二叉树排序树合并到以 t 为头指针的二叉排序树中，

则合并过程为：

1.对 h 树按先序遍历产生初始输入序列；

2.如果 h 所指结点的值小于等于 t 所指结点的值，则插入 t 的左子树中；

3.每个结点都插入成 t 树的新叶子节点；

参考算法代码：

```
void Insert_BSTree(BiTree &t, BiTree h)
{ // 将以 h 为头指针的二叉排序树归并到以 t 为头指针的二叉排序树中去
  BiTree p, lp, rp;
  if ( h ) {
       if (!t )   t = h;
      else if (h->key <= t->key ){   //如果h的根结点小于等于t的根
            rp = h->rchild;
            h->rchild = NULL;
            Insert_BSTree(t->lchild, h); //将除去右子树的h归并到t的左子树
            if (rp)  Insert_BSTree(t, rp); //h的右子树归并到t
        }//if
       else{                          //如果h的根结点小于等于t的根
            lp = h->lchild;
            h->lchild = NULL;
            Insert_BSTree(t->rchild, h); //将除去左子树的h归并到t的左子树
            if (lp)  Insert_BSTree(t, lp);  //h的左子树归并到t
       }//else
   }//if
}//Insert  BSTree
```

www.docin.com

10.7 不难看出，对长度为 n 的记录序列进行快速排序时，所需进行的比较次数依赖于这 n 个元素的初始排列。

1．n=7 时在最好情况下需进行多少次比较？请说明理由。

2. 对 n=7 给出一个最好情况的初始排列实例。

注：1/2 的同学交了第十章作业，其中部分同学对于此题没看清题意，把问题中

的比较次数回答成了趟数，趟数不等于次数。

（1）快速排序的最好情况是指，排序所需的关键字间的比较次数和记录的移动次数最少情况，在 n=7 时，至少需要进行 2 趟排序。因此，当 n=7 在最好的情况下所需进行的比较次数是 10 次。

（2）n=7 时的一个最好的初始排列序列是： 4 7 5 6 3 1 2

10.12 判别以下序列是否为堆（小顶堆或大顶堆）。如果不是，则把它调整为堆（要求记录交换次数最少）。

(1)(100, 86, 48, 73, 35, 39, 42, 57, 66, 21);

(2)(12, 70, 33, 65, 24, 56, 48, 92, 86, 33);

(3)(103, 97, 56, 38, 66, 23, 42, 12, 30, 52, 06, 20);

(4)(05, 56, 20, 23, 40, 38, 29, 61, 35, 76, 28, 100)。

(1)是大顶堆

(2)不是堆，调整为小顶堆（12,24,33,65,33,56,48,92,86,70）

(3)是大顶堆

(4)不是堆，调整为小顶堆(05,23,20,35,28,38,29,61,56,76,40,100)

10.15 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？

算法：

1. 首先 2 个一组比较一轮，较大的加入序列 A，较小的加入序列 B，若剩下一个则同时加入序列 A 和 B；

2. 然后在 A 中求最大值，在 B 中求最小值。

分析：

若 n 为偶数，设 n=2k，则第一步需要 k 次比较，第二步取最大值和最小值各需 k-1 次比较，

共 k+(k-1)+(k-1) = 3k-2 = (3n-4)/2 次；

若 n 为奇数，设 n=2k+1，则第一步需要 k 次比较，第二步取最大值和最小值各需 k 次比较，

共 k+k+k = 3k = (3n-3)/2 次；