1)填空题：序号 1-80

【1，1，2】线性结构中元素之间存在一对一关系，树形结构中元素之间存在__一对多_____关系，图形结构中元素之间存在__多对多_____关系。

【2，1，2】为了最快地存取数据元素，物理结构宜采用_顺序存储___结构。

【3，1，2】数据结构的三要素是____逻辑结构，____物理结构____，__操作____。

【4，1，2】数据的逻辑结构可形式地用一个二元组 B＝(K，R)来表示，其中 K 是__数据元素的有限集合__，R 是__K 上关系的有限集___。

【5，1，2】存储结构可根据数据元素在机器中的位置是否一定连续分为__顺序存储结构__，链式存储结构___。

【6，1，4】度量算法效率可通过__时间复杂度__来进行。

【7，1，4】算法的五个重要特性是确定性、____可行性____、__有穷性_____、输入和输出。

【8，1，4】设 n 为正整数，则下面程序段的时间复杂度是__O(n)__。

```
i=1；k=0；
while(i<n)
{
      k=k+10*i；i++；
}
```

【9，1，4】设 n 为正整数，下面程序段中前置以记号@的语句的频度是____n(n+1)/2____。

```
for (i=0; i<n; i++){
    for (j=0; j<n; j++)
       if (i+j==n-1)
@           a[i][j]=0;
}
```

【10，1，4】设 n 为正整数，试确定下列各程序段中前置以记号@的语句的频度:

(1)
```
i=1;   k=0;
while (i<=n-1){
   i++;
@ k+=10 * i;     // 语句的频度是_____n-1_____。
}
```

(2)
```
k=0;
for (i=1;i<=n;i++){
```

```
        for (j=i; j<=n;j++)
        @     k++;    // 语句的频度是____n(n+1)/2_____。
    }
```

【11，1，4】按增长率由大到小排列下列函数的结果是__ n^2 __ $n\log_2 n$ __ n __ $n^{1/2}$ __ $\log_2 n$ ___ $\log_2(\log_2 n)$,_____。

$\log_2(\log_2 n)$, $n\log_2 n$, n^2, $n^{1/2}$, $\log_2 n$, n

【12，2，1】当线性表的规模比较大，难以估计其存储规模时，一般以采用__动态链表_____的存储结构为好。

【13，2，1】线性表(a_1，a_2，…，a_n)有两种存储结构： 顺序存储结构和链式存储结构，请就这两种存储结构完成下列填充：

__顺序__存储密度较大；__顺序__存储利用率较高；__顺序__可以随机存取；__链式___不可以随机存取；

__链式__插入和删除操作比较方便。

【14，2，2】从一个长度为 n 的顺序表中删除第 i 个元素（1≤i≤n）时，需向前移动___ n-i 个元素。

【15，2，3】带头结点的双链表 L 为空的条件是___ L->next=L 或 L->prior=L ___。

【16，2，3】带头结点的单链表 Head 为空的条件是____ Head->next=NULL _____。

【17，2，3】非空单循环链表 L 中*p 是尾结点的条件是_____p->next=L _____。

【18，2，3】在一个单链表中 p 所指结点(p 所指不是最后结点)之后插入一个由指针 s 所指结点，应执行 s->next=__p->next ___;和 p->next=_____s_____的操作。

【19，2，3】在一个单链表中的指针 p 所指结点之前插入一个由指针 s 所指结点，可执行以下操作序列：

```
s->next=____p->next；____;
p->next=s;
t=p->data;
p->data=_____s->data;_____;
s->data=t;
```

【20，2，3】在一个单链表中删除 p 所指结点时，应执行以下操作：

```
q= p->next;
p->data= p->next->data;
p->next=____p->next->next _____;
free(q);
```

【21，2，3】在单链表中，删除指针 P 所指结点的后继结点的语句是___P->next = P->next->next;___。

【22，2，3】带头结点的单循环链表 Head 的判空条件是__Head->next == Head;___； 不带头结点的单循环链表的判空条件是___Head == NULL;__。

【23，2，3】删除带头结点的单循环链表 Head 的第一个结点的操作是___Head->next = Head->next->next;___；删除不带头结点的单循环链表的第一个结点的操作是__Head = Head->next;___。

【24，2，3】已知 L 是带表头结点的非空单链表，且 P 结点既然不首元结点,也不是尾元结点,试从下列提供的答案中选择合适的语句序列。

a. 删除 P 结点的直接前驱结点的语句序列是_____10 12 8 11 4 14_____。

b. 删除结点 P 的语句序列是_____10 12 7 3 14_____。

c. 删除尾元结点的语句序列是_____9 11 3 14_____。

(1) P = P->next;
(2) P->next = P;
(3) P->next = P->next ->next;
(4) P = P->next ->next;
(5) **while** (P != **NULL**) P = P->next;
(6) **while** (Q->next != **NULL**){P = Q; Q = Q->next};
(7) **while** (P->next != Q) P = P->next;
(8) **while** (P->next->next != Q) P = P->next;
(9) **while** (P->next->next != **NULL**) P = P->next;
(10) Q = P;
(11) Q = P->next;
(12) P = L;
(13) L = L->next;
(14) **free** (Q);

【25，3，1】栈操作的原则是____先进后出/后进先出____。

【26，3，1】对一个栈，给定输入的顺序是 A、B、C，则全部不可能的输出序列有__不可能得到的输出序列有 CAB_____。

【27，3，1】数据 A、B、C、D 依次进栈后，再从栈中取一数据，则它是___D。则本栈得到 DCBA 的输出序列，其理由是___根据后进先出的原则，首先得到 D，在栈内的数由底到顶依次是 A、B、C，而出栈的次序刚好相反，即 DCBA。___。

【28，3，1】.在栈顶指针为 HS 的链栈中，判定栈空的条件是 head->next==NULL_____。

【29，3，1】将递归算法改写成等价的非递归算法，通常应该设置___堆栈___的数据结构

【30，3，2】下列程序把十进制数转换为十六进制数，请填写合适的语句成分。（每空 2 分）

```
void conversion10_16()
{   InitStack(&s);
    scanf("%d",&N);
    while(N){
        ①  _____Push(s, N%16)_____;
        N = N/16;
    }
    while(!StackEmpty(s)){
        ②  _____Pop(s, e)_____;
        if(e<=9)printf("%d",e);
        else printf("%c",e-10+'A');
    }
}  /* conversion */
```

【31，3，4】若一个栈的输入序列为 1,2,3,…,n，输出序列的第一个元素为 n，则第 i 个输出元素是__n－i＋1__。

【32，3，4】若用一个大小为 6 个元素的数组来实现循环队列，且当前 rear=0 和 front=3。当从队列中删除一个元素，再加入两个元素后，rear 和 front 的值分别是___2____和__4_____。

【33，3，4】已知一个栈的输入序列为 1，2，3，…，n,输出序列为 $a_1,a_2,a_3,\ldots,a_n$，那么 a_2=n 的输出序列共有___n-1___种。

【34，3，4】堆栈和队列都是线性表，堆栈是_____后进先出_____的线性表，而队列是_____先进先出_____的线性表。

【35，3，4】从循环队列中删除一个元素时，其操作是____先移动队首元素,后取出元素_____。

【36，3，4】若用一个大小为 6 个元素的数组来实现循环队列，且当前 rear=0 和 front=3。当从队列中删除一个元素，再加入两个元素后，rear 和 front 的值分别是_____2_____和4_____。

【37，3，4】下面是关于循环队列的操作，请在划线空白处填写合适语句成分。

```
Status EnQueue(SqQueue  &Q, QelemType e)
{

if((Q.rear+1)%MAXSIZE==Q.front)_____) retrun
ERROR;
        Q.base[(Q.rear)] = e;
```

```
        Q.rear=(Q.rear+1)%MAXSIZE ;
        return OK;
    } // EnQueue
```

【38，4，1】空串是 零个字符的串 ，其长度为 零 。

【39，4，1】设串 s1=”teachers and “,s2=”students”, 则 StrLength(s2)= 8 ；Concat(s1,s2)= ”teachers and students” 。

【40，4，1】设 s = ‘I AM A TEACHER’，其长度是 14 。

【41，4，1】两个串相等的充分必要条件是 两个串的长度相等且对应位置的字符相同 。

【42，4，2】串的两种最基本的存储方式是 顺序存储方式和链接存储方式 。

【43，4，3】令有串 u=”aabcaab”和 v=”abcaabcaabcaaba”,

(1) 求 Index(v,u,5)的值：Index(v,u,5)= 8 ; (2 分)

(2) 求出 u 作为模式串时在 KMP 算法中的 next[j]值。(2 分)

j	1	2	3	4	5	6	7
u	a	a	b	c	a	a	b
next[j]	0	1	2	1	1	2	3

【44，5，1】二维数组 A[10][20]采用列序为主方式存储，每个元素占一个存储单元，并且 A[0][0]的存储地址是 200，则 A[6][10]的地址是 332 。

【45，5，1】设每个元素需要 8 个字节，顺序存储 100 个元素,若首地址是 2500，那么第 50 个元素的地址是 2892 。

【46，5，2】已知二维数组 A[m][n]采用行序为主方式存储，每个元素占 k 个存储单元，并且第一个元素的存储地址是 LOC(A[0][0])，则 A[i][j]的地址是 LOC (A[0][0]) + (n*i + j) k

【47，5，2】C 语言采用行优先方式存放数组元素，数组下标从 0 开始。设维数为（5,6,7）的数组 A_{5x6x7} 的起始存储地址为 Loc[0][0][0]=1000，每个数组元素占用 4 个字节。则元素 A[4][4][4]所在的地址 Loc[4][4][4]= 1800 。

【48，5，2】按行优先次序列出三维数组 A[2][3][2]的所有 12 个元素在内存中的存储次序，它们依次是：

A[0][0][0]
A[0][0][1]
A[0][1][0]
A[0][1][1]
A[0][2][0]

A[0][2][1]
A[1][0][0]
A[1][0][1]
A[1][1][0]
A[1][1][1]
A[1][2][0]
A[1][2][1]

【49，5，3】对一个 10 阶对称矩阵 A，采用压缩存储方式（以行序为主序，且 A[0][0]的地址为 1）,则 A[8][5]的地址是____34____。

【50，5，3】对一个 10 阶三对角矩阵 A，采用压缩存储方式（以行序为主序，且 A[0][0]的地址为 1，每个元素占 4 个字节）,则 A[6][5]的地址是____69____。

【51，5，3】对一个对称矩阵 A（$a_{ij}=a_{ji}$,0<=i,j<=n），采用下三角压缩存储方式存储在一维数组 S[1..M]中（以行序为主序，且 A[0][0]=S[1]），则 A[i][j]（i>=j）对应 S 中的下标是____i*(i+1)/2 + j+1____；一维数组 S 的大小 M 至少为____(n+1)*(n+1)____。

【52，6，1】
已知一棵树边的集合是{<a,d>,<d,c>,<d,j>,<e,a>,<f,g>,<d,b>,<g,h>,<g,i>,<e,f>}。那么根结点是____e____，结点 b 的双亲是____d____，结点 a 的子孙有____bcdj____，树的深度是____4____，树的度是____3____，结点 g 在树的第____3____层。

【53，6，2】通常使用____二叉链表____来表示二叉树结构。

【54，6，2】从概念上讲，树与二叉树是二种不同的数据结构，将树转化为二叉树的基本的目的是____.树可采用二叉树的存储结构并利用二叉树的已有算法解决树的有关问题____。

【55，6，2】在图 1 至图 5 中，____1,2,3,4,5____是树，____1,2,3,4____是二叉树，____1,2,3____是完全二叉树，____1,3____是满二叉树。

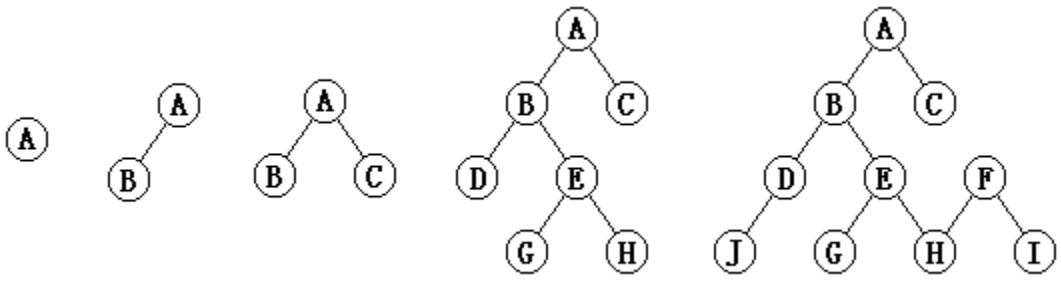


图 1　　图 2　　图 3　　图 4　　图 5

【56，6，3】在图 4 中，结点 H 在这棵树的前序、中序和后序遍历次序中分别是___6__、第__5___和第___3___个结点。

【57，6，2】满三叉树的第 i 层的结点个数为____3^{i-1}____，深度为 h 时该树中共有____3^h-1结点。

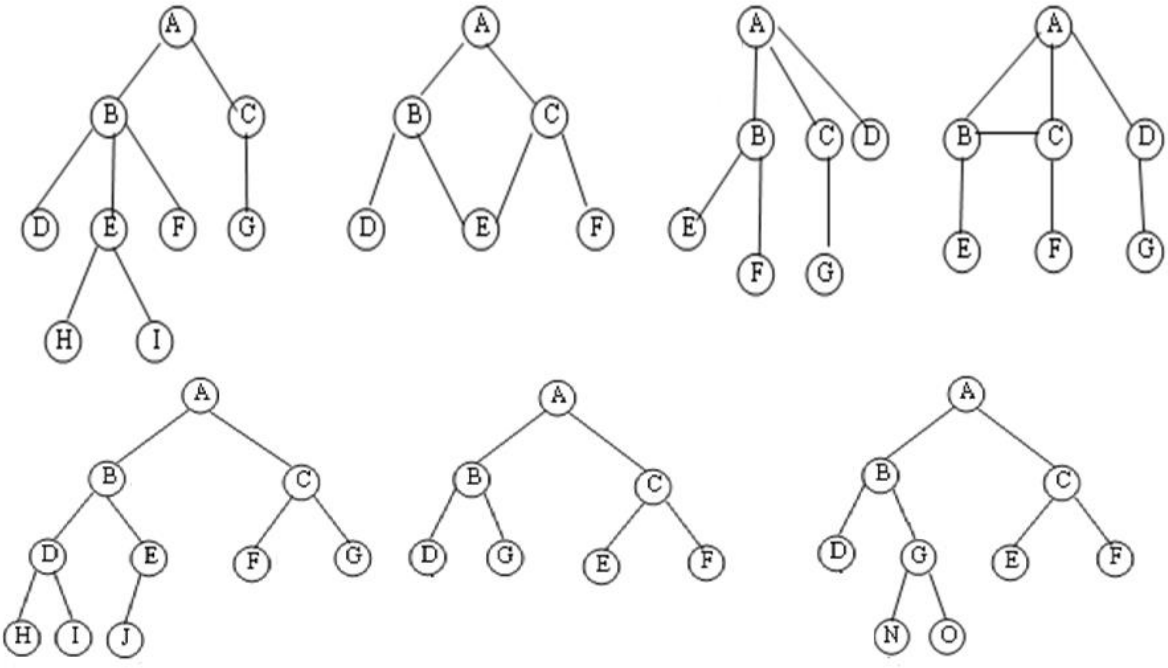
【58，6，2】在图 4 中，A 是___根___结点，D 是___叶子___结点，B 是 E 的___父亲_____，B 是 G 的____祖先_____,D 是 E 的__兄弟_____。这棵树的度是___6_____,深度是__4_____。

【59，6，2】程序填空：下列算法是求以二叉链表存储的二叉树中的最小值，设数据域的类型为 int。

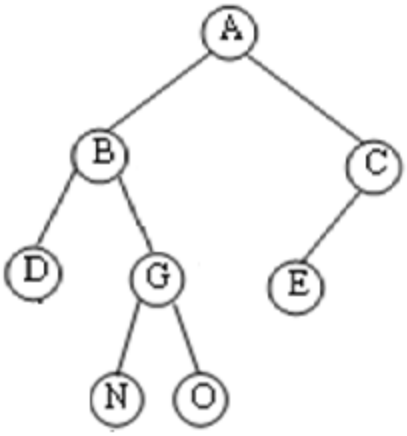
```
void minnode(BiTree T, int *min)
{
    if(T!=NULL)
    {
        if(___*min>T->data___) *min = T->data;
        minnode(T->lchild,min);
        ___minnode(T->rchild,min)___;
     }
}
```

【60，6，2】已知一棵完全二叉树有 56 个叶子结点，从上到下、从左到右对它的结点进行编号，根结点为 1 号。则该完全二叉树总共结点有___111_____个；有____7___层；第 91 号结点的双亲结点是___45____号；第 63 号结点的左孩子结点是_____号。

【61，6，2】下列表示的图中，共有__5____个是树；有___3____个是二叉树；有__2____个是完全二叉树。



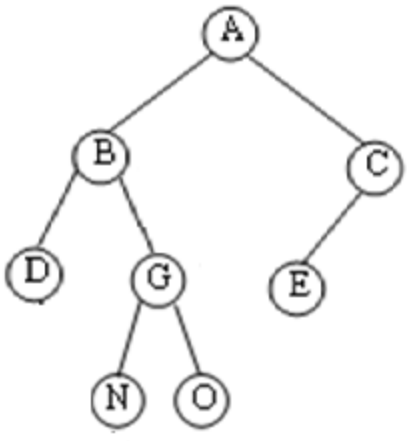
【62，6，3】下列二叉树的中序遍历序列是____DBNGOAEC_____;后序遍历序列是_____DNOGBECA_____。



【63，6，3】一棵二叉树的中序遍历序列是 DBNGOAEC,后序遍历序列是 DNOGBECA，则其先序遍历的序列中的第一个元素是__A___，第五个元素是_N____，最后一个元素是_E_

ABDGNOCE

【64，6，3】下列二叉树的先序遍历序列的第 5 个结点是___N___；第 8 个结点是__E___；
ABDGNOCE
后序遍历序列的第 2 个结点是_N____；第 6 个结点是__E___。DNGOBECA



【65，6，3】如果某二叉树的后序遍历序列是 ABCDEFGHI，中序遍历序列是 ACBIDFEHG，则其先序遍历序列的第一个字母是___I___，最后一个字母是____G____。

【66，6，3】程序填空：设算法 DFS(Mgraph *G, int i)是无向图 G 从 i 顶点开始的深度优先遍历算法。下列算法是判断无向图 G 是否是连通的。

```
int isconnect(Mgraph *G)
{       int i,k=0;
        for(i=0; i<G->vexnum; i++)
              visited[i] = 0;
        for(i=0; i<G->vexnum; i++)
              if(!visited[i])
            {    ____k++____;
                 DFS(G, i);
            }
        if(k==1)  ____return 1____;
        else      return 0;
}
```

【67，7，2】图有____邻接矩阵____和____邻接表____等存储结构。

【68，7，2】设无权图 G 的邻接矩阵为 A，若（vi,vj）属于图 G 的边集合，则对应元素 A[i][j]等于__1__，设无向图 G 的邻接矩阵为 A，若 A[i][j]等于 0，则 A[j][i]等于____0____。

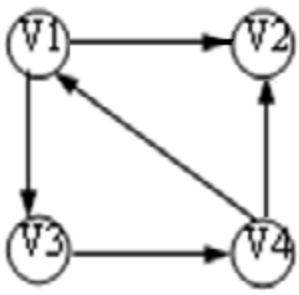
【69，7，2】若一个图用邻接矩阵表示，则计算第 i 个结点的入度的方法是____求矩阵第 i 列非零元素之和____。

【70，7，2】若一个图用邻接矩阵表示，则删除从第 i 个顶点出发的所有边的方法是____矩阵第 i 行全部置为零____。

【71，7，4】n 个顶点的连通图至少有___n－1__条边。

【72，7，4】设一个图 G={V,{A}},V={a,b,c,d,e,f},A={<a,b>,<b,e>,<a,e>,<c,a>,<e,d>,<d,f>,<f,c>}。那么顶点 e 的入度是__2____；出度是____1____；通过顶点 f 的简单回路有____2____条；就连通性而言，该图是__强连通____图；它的强连通分量有____1____个；其生成树可能的最大深度是__5____。

【73，7，5】下面有向图共有____4___个顶点；从 v3 到 v2 的最短简单路径之一是___v3—v4—v2_____；v1 的出度是___2_____；包含所有顶点的简单路径之一是_____v3—v4—v1—v2_____。



【74，9，2】n 个结点的二叉排序树的最大深度是__n____，最小深度为____$[\log_2 n]+1$____

【75， 9，3】设有一组关键字{9,01,23,14,55,20,84,27}，采用哈希函数：H(key)=key mod 7，表长为 10，用开放地址法的二次探测再哈希方法 Hi=(H(key)+di) mod 10 (di = 1^2,2^2,3^2,…,)解决冲突。要求：对该关键字序列构造哈希表，并计算查找成功的平均查找长度。

哈希地址	0	1	2	3	4	5	6	7	8	9
关键字	14	01	9	23	84	27	55	20		
比较次数	1	1	1	2	3	4	1	2		

平均查找长度：ASL = (1+1+1+2+3+4+1+2)/8 = 15/8
以关键字 27 为例，H(27)=27%7=8(冲突)，H1=(6+1)%10=7(冲突)，H2=(6+2^2)%10=0(冲突)，H3=(6+3^3)%10=5，所以比较了 4 次。

【76，10，1】排序过程一般需经过两个基本操作，它们是___比较___和___移动___。

【77，10，2】结点的关键字序列是（F,B,J,G,E,A,I,D,C,H），对它按字母的字典序进行排列。如果采用 Shell 排序方法，那么步长取 5 时，第一次扫描结果的前 5 个字母分别是（A, B, D, C, E）__。

【78，10，2】在对一组关键字是（54,38,96,45,15,72,60,23,83）的记录进行直接插入排序时，当把第七个记录（关键字是 60）插入到有序表时，为寻找插入位置需比较__3__次。

【79，10，3】在利用快速排序方法对一组关键字是（54,38,96,45,15,72,60,23,83）的记录进行排序时，需要递归调用 partition 函数，递归的最大深度（设第一次调用深度为 1）为___3___，共需 5 次递归调用，其中第二次递归调用是对关键字是___（23，38，15，45）___的一组记录进行的。

【80，10，4】插入排序、希尔排序、选择排序、快速排序、堆排序、归并排序、和基数排序方法中，不稳定的排序方法有___希尔排序、快速排序、堆排序__。

4) 分析计算作图题：序号 1-55（选自《数据结构题集》—严蔚敏等编）

【1，　1，4】（选自《数据结构题集》1.8，选做题）

设 n 为正整数，试确定下列各段程序中前置以记号@的语句的频度(语句的频度指的是该语句重复执行的次数)。

```
(1)   i=1;  k=0;
      while (i<=n-1){
            i++;
        @   k+=10*i;
      }
```

答：该语句的频度为 n-1

```
(2)   x=91;   y=100;
      while(y>0){
      @   if(x>100) {x-=10; y--;}
                else x++;
      }
```

答：在循环中，if 语句为真 1 次，else 语句执行 10 次，所以 if 语句执行 11 次 y 的值减 1。该语句的频度为 100x(1+10) = 1100

【2，　1，4】（选自《数据结构题集》1.9，选做题）

假设 n 为 2 的乘幂，并且 n>2，试求下列算法的时间复杂度及变量 count 的值（以 n 函数形式表示）

```
int Time (int n) {
    count =0; x=2;
    while( x<n/2 )   {
          x*=2;   count++;
    }
    return (count);
}// Time
```

答：从 while 的循环控制可以看出，n 每增加一倍,count 的值增加 1。所以该算法的时间复杂度为 $O(\log_2 n)$，根据初值，变量 count 的值为「$\log_2(n-2)$」- 1

【3，　2，3】（选自《数据结构题集》2.6，必做题）

已知 L 是无表头结点的单链表，且 P 既不是首元结点，也不是尾元结点，试从下列提供的答案中选择合适的语句序列。

a．在 P 结点后插入 S 结点的语句序列是　（4）（1）

b．在 P 结点前插入 S 结点的语句序列是　（7）（11）（8）（4）（1）

c．在表首插入 S 结点的语句序列是　（5）（12）

d．在表尾插入 S 结点的语句序列是　（9）（1）（6）

（1）P —>next ＝ S ；

（2）P —>next ＝ P —>next —>next；
（3）P —>next ＝ S —>next；
（4）S —>next ＝ P —>next；
（5）S —>next ＝ L；
（6）S —>next ＝ NULL；
（7）Q ＝ P；
（8）while （P —>next ！＝ Q）P ＝ P —>next；
（9）while （P —>next ！＝ NULL） P ＝ P —>next；
（10）P ＝ Q；
（11）P ＝ L；
（12）L ＝ S；
（13）L ＝ P；

【4， 2，2】（选自《数据结构题集》2.10，选做题）

指出以下算法中的错误和低效（即费时）之处，并将它改写为一个既正确又高效的算法。

```
Status DeleteK（SqList &a，int i，int k）{
  //本过程从顺序存储结构的线性表 a 中删除第 i 个元素起的 k 个元素
   if（i < 1 || k < 0 || i + k > a.length）return INFEASIBLE；// 参数不合法
   else {
      for（count ＝ 1；count < k；count ++）{
         //删除一个元素
         for（j ＝ a.length；j >＝ i +1；j －－）a.elem[j－1] ＝ a.elem[j]；
         a. length －－；
      }
   return OK；
} // DeleteK
```

答：错误之处：题中有下划线处应改为：

for （j ＝ i+1；j<= a.length；j++） a.elem[j－1] ＝ a.elem[j]；

低效费时之处：该算法每删除一个元素，其后所有的元素都向前移一位，包括将要删除的元素，比较耗时间。（其中 n＝a.length）

下划线的语句的频度为：

$$n-i+(n-i-1)+\cdots+n-i-k+1=k(n-i)-k(k-1)/2$$

改进方法是将 k 个元素在一个循环内删除，算法如下：

```
Status DeleteK（SqList &a，int i，int k）{
//本过程从顺序存储结构的线性表 a 中删除第 i 个元素起的 k 个元素
    if（i < 1 || k < 0 || i + k > a.length）return INFEASIBLE；// 参数不合法
    else {
       for（count ＝ 0；count < a.length－k－i；count ++）
           a.elem[i-1+ count] ＝ a.elem[k + i-1 + count ]；
        //删除 k 个元素
       a.length ＝ a.length－k；
```

```
        }
    return OK;
} // DeleteK
```

【5，　3，1】（选自《数据结构题集》3.4，必做题）

简述以下算法的功能（栈的元素类型 SElemType 为 int）

```
（1）Status  algo1（Stack  S）{
          int  i，n，A[255]；
          n ＝ 0；
          while  （！StackEmpty（S））{n ++；Pop（S，A[n]）；}
          for（i ＝ 1；i<＝ n；i++）Push（S，A[i]）；
    }
```

功能：利用数组 A，将栈中所有的元素倒置。

```
（2）Status  algo2（Stack S，int e）{
          Stack T；int d；
          InitStack（T）；
          while（！StackEmpty（S））{
              Pop（S，d）；
              if（d！＝e）Push（T，d）；
          }
          while（！StackEmpty（T））{
              Pop（T，d）；
              Push（S，d）；
          }
    }
```

功能：将栈 S 中与 e 相同的元素删除。

【6，　3，1】（选自《数据结构题集》3.15，选做题）

假设以顺序存储结构实现一个双向栈，即在一堆数组的存储空间中存在着两个栈，它们的栈底分别设在数组的两个端点。试编写实现这个双向栈 tws 的三个操作：初始化 inistack（tws）、入栈 push（tws，i，x）和出栈 pop（tws，i）的算法，其中 i 为 0 或 1，用以分别指示设在数组两端的两个栈，并讨论按过程（正/误状态变量可设为变参）或函数设计这些操作算法各有什么优缺点。

```
typedef   struct {
      SElemType   *base[2];
      SElemType   *top[2];
      int   stacksize;
} SqStack;
Status   initstack ( SqStack &tws) {      // 初始化，构造一个空栈 tws
      tws．base[0] ＝（SElemType *）malloc（STACK_INIT_SIZE * sizeof（））；
      if（！tws.base[0]）exit（OVERFLOW）；  // 存储分配失败
      tws．stacksize ＝ STACK_INIT_SIZE；
      tws．base[1] ＝ tws．base[0] + tws.stacksize－1；
```

```
    tws. top[0] = tws. base[0];
    tws. top[1] = tws. base[1];
    return  OK;
} // initstack

Status  push（SqStack &tws，int i，ElemType x） {  // 插入元素 x 为新的栈顶元素
    if（tws.top[0] － base[0] + tws.top[1] － base[1] >= tws.stacksize） {
    // 栈满，追加存储空间
    increment0 = tws.top[0] － base[0];  // 0 端的元素个数
    increment1 = tws.top[1] － base[1];   // 1 端的元素个数
    tws. base[0] =（SElemType *）realloc（tws.base[0] ,(tws.base[0]+
      STACKINCREMENT) * sizeof（ElemType））;
    if（！tws.base[0]）exit（OVERFLOW）;  // 存储分配失败
    tws.stzcksize += STACKINCREMENT;
    temp=tws.base[1];
    tws. base[1] = tws. base[0] + tws.stacksize－1;
    for(i=0; i<increment1; i++)
        *(tws.base[1] - i) = *(temp - i);  //把 1 端栈中元素移到新的双向栈的 1 端
    tws. top[0] = tws. base[0] + increment0;
    tws. top[1] = tws. base[1] - increment1;
    }
    if（i） * tws.top[1] -- = x;
    else
          * tws.top[0] ++ = x;
    return OK;
} // push

Status  pop（SqStack &tws，int i） {
    // 若栈不空，则删除 tws 的栈顶元素，并返回 OK；否则返回 ERROR
    if（tws.top[0] == base[0] || tws.top[1] == base[1]） return ERROR;
    if（i） ++ tws.top[1];
    else
        －－tws.top[0];
    return OK;
} // pop
```

【7，　3，4】（选自《数据结构题集》3.13，必做题）

简述以下算法的功能（栈和队列的元素类型均为 int）

```
void algo3(Queue &Q){
 Stack S; int d;
 InitStack (S);
 while (!QueueEmpty (Q)) {
     DeQueue (Q, d); Push (S,d);
 }
```

```
    while (!StackEmpty (S)) {
        Pop (S, d); EnQueue(Q, d);
    }
}
```

答：将队列 Q 中的元素变成倒置排列。

【8， 3，2】（选自《数据结构题集》3.19，选做题）

假设一个算术表达式中可以包含三种括号：圆括号“(”和“)”，方括号“[”和“]”和花括号“{”和“}”，且这三种括号可以按任意的次序嵌套使用（如：…[…{…}…[…]…]…[…]…(…)…）。编写判别给定表达式所含括号是否正确配对出现的算法（已知表达式已存入数据元素为字符的顺序表中）。

解：算法如下：

```
Status MatchBrackets ( ) {
    SqStack &S;
    InitStack (S);
    while ((c=getchar())!= '\n' ){
        if(!StackEmpty (S) && (c== ')' && GetTop (S)== '(' ||
                               c== ']' && GetTop (S)== '[' ||
                               c== '}' && GetTop (S)== '{' )     )
            Pop (S);    //判断括号是否匹配
        else
            if(IsLeftBracket(c)) Push (S, c);   //判断是否是括号
    }
    if(!StackEmpty(S))   return ERROR;
    else    return TRUE;
}
```

【9， 4，1】（选自《数据结构题集》4.3，必做题）

设 s = 'I AM A STUDENT' ,t = 'GOOD' ,q = 'WORKER' 。求：

StrLength(s), StrLength(t), SubString(s,8,7), SubString(t,2,1), Index(s, 'A'), Index(s, t), Replace(s, 'STUDENT' ,q), Concat(SubString(s,6,2),Concat(t,SubString(s,7,8)))

答：StrLength(s) = 14, StrLength(t) = 4,
SubString(s,8,7) = 'STUDENT', SubString(t,2,1) = 'O',
Index(s, 'A') = 3, Index(s, t) = 0,
Replace(s, 'STUDENT' ,q) = 'I AM A WORKER',
Concat(SubString(s,6,2),Concat(t,SubString(s,7,8)))
= 'A GOOD STUDENT'

【10， 5，1】（选自《数据结构题集》5.1，必做题）

假设有两维数组 $A_{6\times8}$，每个元素用相邻的 6 个字节存储，存储按字节编址。已知 A 的起始地址为 1000，计算：

(1) 数组 A 的存储量； **= 6x8x6 = 288 (字节)**

(2) 数组 A 的最后一个元素 a_{57} 的第一个字节的地址；**=1000+288-6=1282**

(3) 按行存储时，元素 a_{14} 的第一个字节的地址；**=1000+(8x1+4)x6=1072**

(4) 按列存储时，元素 a_{47} 的第一个字节的地址；**=1000+(7x6+4)x6=1276**

【11， 6，1】（选自《数据结构题集》6.1，必做题）

已知一棵树边的集合为{<I，M>，<I，N>，<E，I>，<B，E>，<B，D>，<A，B>，<G，J>，<G，K>，<C，G>，<C，F>，<H，L>，<C，H>，<A，C>}，请画出这棵树，并回答下列问题：

（1）哪个是根结点？

（2）哪些是叶子结点？

（3）哪个是结点 G 的双亲？

（4）哪些是结点 G 的祖先？

（5）哪些是结点 G 的孩子？

（6）哪些是结点 E 的子孙？

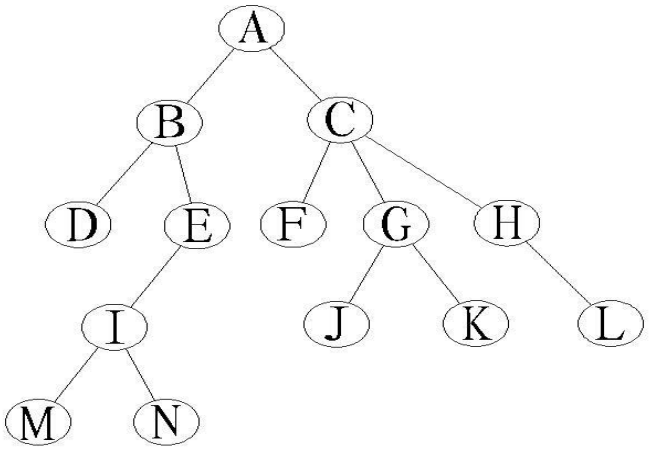
（7）哪些是结点 E 的兄弟？哪些是结点 F 的兄弟？

（8）结点 B 和 N 的层次号分别是什么？

（9）树的深度是多少？

（10）以结点 C 为根的子树的深度是多少？

解：这棵树为:



所以，根结点为 A；

叶子结点是 D，M，N，F，J，K，L；

结点 G 双亲为 C；

结点 G 的祖先为 A，C；

结点 E 的子孙为 I，M.，N；

结点 E 的兄弟有 D，结点 F 的兄弟有 G，H；

结点 B 和 N 的层次号分别是 2，5；

树的深度是 5；

以结点 C 为根的子树的深度为 3。

【12， 6，1】（选自《数据结构题集》6.4，选做题）

一棵深度为 H 的满 k 叉树有如下性质第 H 层上的结点都是叶子节点，其余各层上每个结点都有 k 棵非空子树。如果按层次顺序从 1 开始对全部结点编号，问：

（1）各层的结点数目是多少？

（2）编号为 p 的父结点（若存在）的编号是多少？

（3）编号为 p 的结点的第 i 个儿子结点（若存在）的编号是多少？
（4）编号为 p 的结点有右兄弟的条件是什么？其右兄弟的编号是多少？

解：（1）各层的结点数为k^{n-1}（n＝1，2，…，H）；

（2）首先，p＝1 时，没有父结点，$p \neq 1$时，通过归纳可得，编号为 a 的结点，它的所有子结点的编号为从$k(a-1)+2$到$ka+1$，则编号为 p 的结点的父结点为$\lceil \frac{p-2+k}{k} \rfloor$；

（3）由上可得，第 i 个儿子结点的编号为$k(p-1)+2+i-1$；

（4）p 有右兄弟，即 p 不为最右的结点，条件为$(p-1)\%k \neq 0$，右兄弟的编号为 p＋1。

【13， 6，2】（选自《数据结构题集》6.5，选做题）

已知一棵度为 k 的树中有n_1个度为 1 的结点，n_2个度为 2 的结点，…，n_k个度为 k 的结点，问该树中有多少个叶子结点？

解：设结点总数为 n，则有 $n= n_0+n_1+n_2+...+n_k$，
因为除了根结点外，所有结点都有边进入，所以有 n=B+1,而
$B = 1*n_1+2*n_2+...+k*n_k = \sum i*n_i$,所以有
$n_0+n_1+n_2+...+n_k= \sum i*n_i+1$
$n_0= \sum (i-1)*n_i+1$

【14， 6，3】（选自《数据结构题集》6.13，必做题）

假设 n 和 m 为二叉树中的两结点，用“1”、“0”、“Φ”（分别表示肯定、恰恰相反或者不一定）填写下表：

已知 \ 问 答:	前序遍历时 n 在 m 前？	中序遍历时 n 在 m 前？	后序遍历时 n 在 m 前？
n 在 m 左方	1	1	1
n 在 m 右方	0	0	0
n 是 m 祖先	1	Φ	0
n 是 m 子孙	0	Φ	1

注：如果（1）离 a 和 b 最近的共同祖先 p 存在，且（2）a 在 p 的左子树中，b 在 p 的右子树中，则称 a 在 b 的左方（即 b 在 a 的右方）

【15， 6，3】（选自《数据结构题集》6.14，选做题）

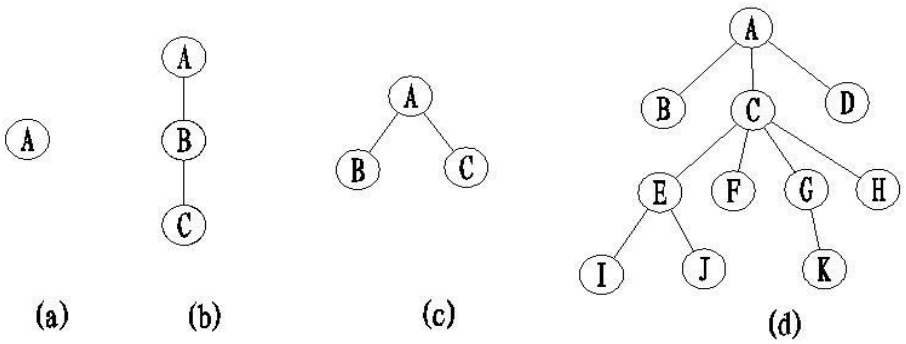
找出所有满足下列条件的二叉树：

（a）它们在先序遍历和中序遍历时，得到的结点访问序列相同；
（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，得到的结点访问序列相同；

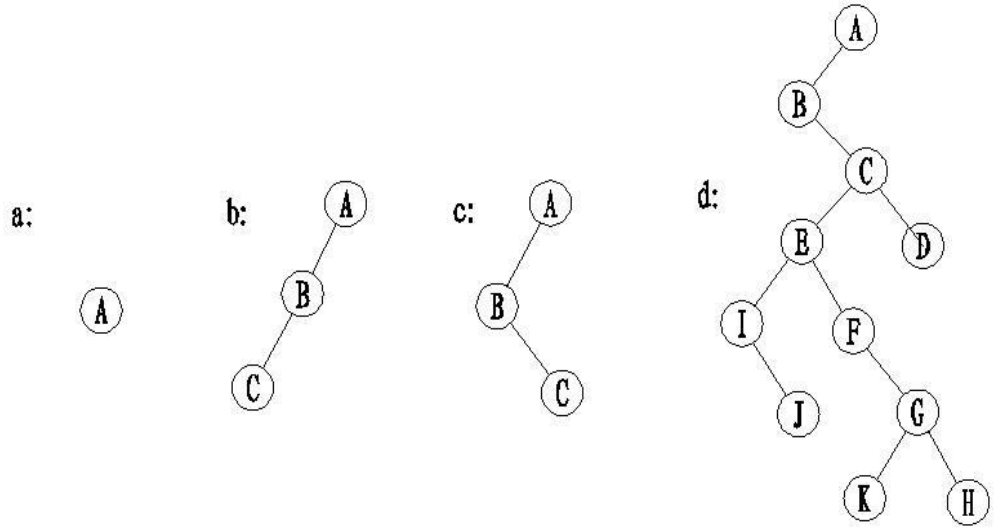
答：（a）除叶子结点外所有结点都只有右孩子的树或只有根结点的树；
（b）只有根结点以及除叶子结点外所有结点都只有左孩子的树；
（c）只有根结点的树。

【16， 6，4】（选自《数据结构题集》6.19，必做题）

分别画出和下列树对应的各二叉树：



答：用孩子兄弟表示法将树表示成二叉树，二叉树的左右结点分别为第一个孩子结点和下一个兄弟结点，表示如下：



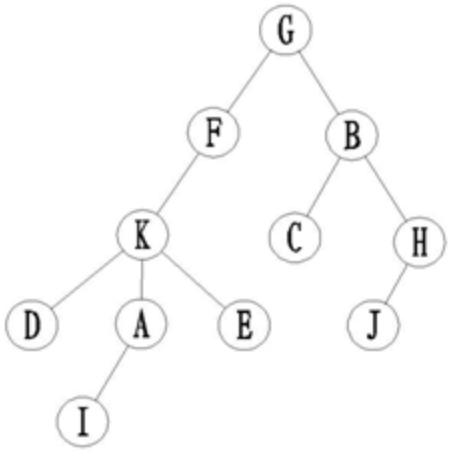
【17， 6，3】（选自《数据结构题集》6.23，选做题）

画出和下列已知序列对应的树 T

树的先根次序访问序列为 GFKDAIEBCHJ；

树的后根次序访问序列为 DIAEKFCJHBG。

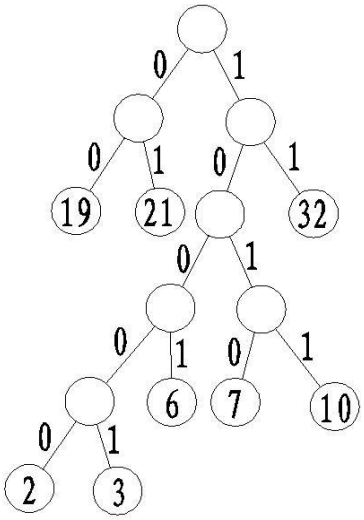
解：对应的树如下：



【18， 6，5】（选自《数据结构题集》6.26，必做题）

假设用于通信的电文仅有 8 个字母组成，字母在电文中出现的频率分别为 0.07，0.19，0.02，0.06，0.32，0.03，0.21，0.10.试为这 8 个字母设计哈夫曼编码。使用 0—7 的二进制表示形式是另一种编码方案。对于上述实例，比较两种方案的优缺点。

解：哈夫曼编码树如下：



所以编码如下：

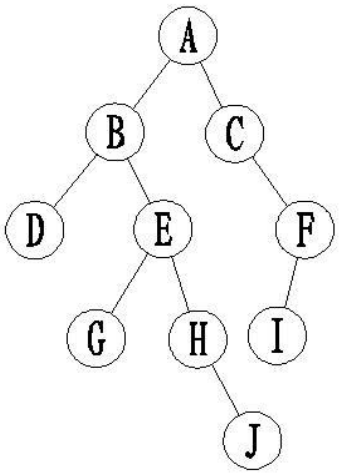
字母	0.07	0.19	0.02	0.06	0.32	0.03	0.21	0.10
编码	1010	00	10000	1001	11	10001	01	1011

带权路径长度 WPL＝0.19×2＋0.21×2＋0.02×5＋0.03×5＋0.06×4＋0.07×4＋0.10×4＋0.32×2＝2.61，而采用等长编码时，每个字母至少 3 位，故带权路径长度 WPL＝3.采用哈夫曼编码可以大大提高通信信道的利用率。

【19， 6，2】（选自《数据结构题集》6.29，必做题）

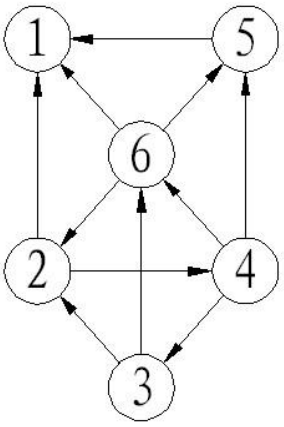
假设一棵二叉树的层序序列为 ABCDEFGHIJ 和中序序列为 DBGEHJACIF。请画出该树。

解：该二叉树如下：



【20， 7，2】（选自《数据结构题集》7.1，必做题）
已知如右图所示的有向图，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。

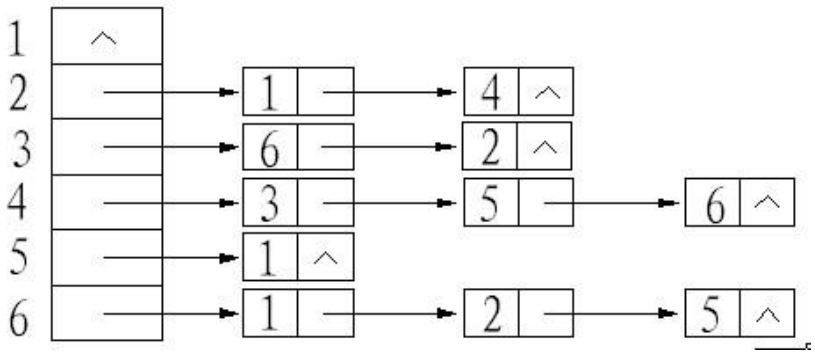


答：（1）各顶点的入/出度如下：
顶点 1：3/0；顶点 2：2/2；顶点 3：1/2；顶点 4： 1/2；
顶点 5：2/1；顶点 6：2/3。

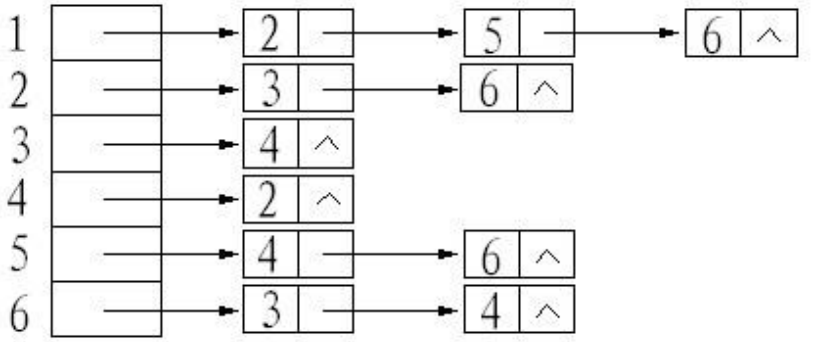
（2）邻接矩阵如下：

	1	2	3	4	5	6
1	0	0	0	0	0	0
2	1	0	0	1	0	0
3	0	1	0	0	0	1
4	0	0	1	0	1	1
5	1	0	0	0	0	0
6	1	1	0	0	1	0

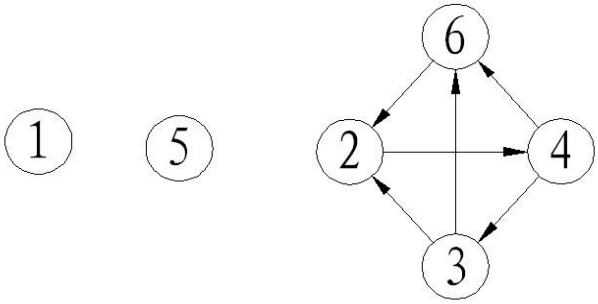
（3）邻接表如下：



（4） 逆邻接表如下：

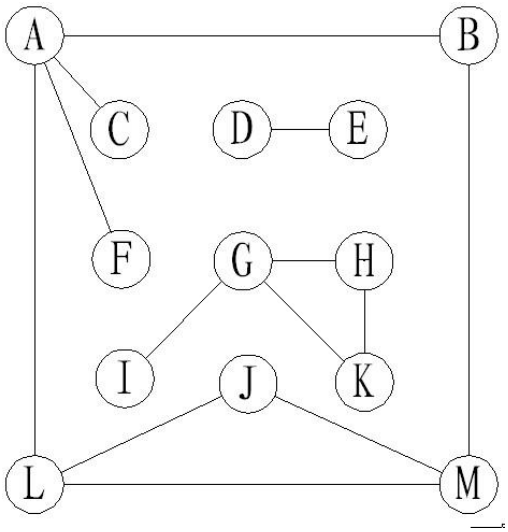


（5） 有以下 3 个强连通分量

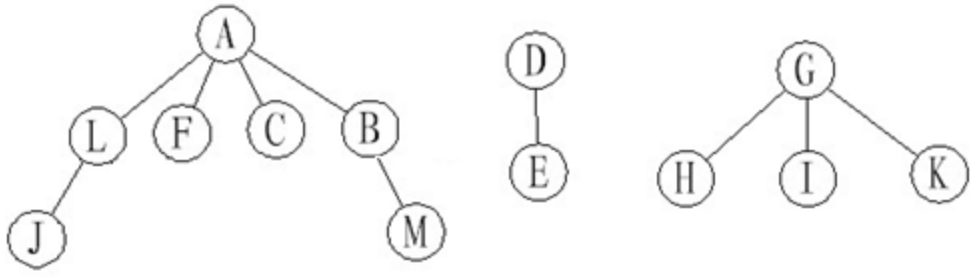


【21， 7，3】（选自《数据结构题集》7.4，选做题）

试对教科书 7.1 节中图 7.3（a）所示的无向图(如下图)，画出其广度优先生成森林。



答：其广义优先生成森林如下：

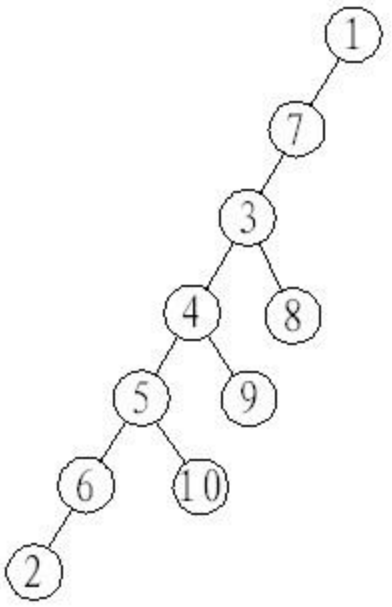


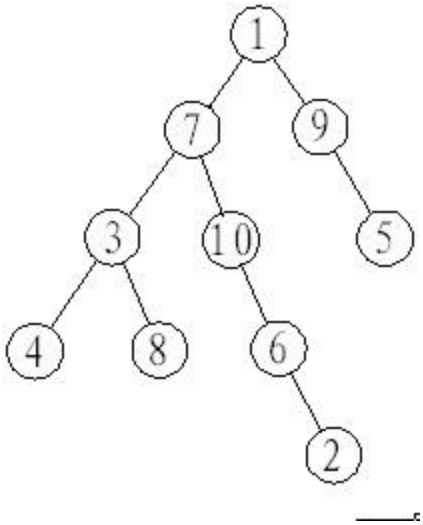
【22， 7，4】（选自《数据结构题集》7.5，选做题）

已知以二维数组表示的图的邻接矩阵如下图所示。试分别画出自顶点 1 出发进行遍历所得的深度优先生成树和广度优先生成树。

	1	2	3	4	5	6	7	8	9	10
1	0	0	0	0	0	0	1	0	1	0
2	0	0	1	0	0	0	1	0	0	0
3	0	0	0	1	0	0	0	1	0	0
4	0	0	0	0	1	0	0	0	1	0
5	0	0	0	0	0	1	0	0	0	1
6	1	1	0	0	0	0	0	0	0	0
7	0	0	1	0	0	0	0	0	0	1
8	1	0	0	1	0	0	0	0	1	0
9	0	0	0	0	1	0	1	0	0	1
10	1	0	0	0	0	1	0	0	0	0

答：其深度优先生成树和广度优先生成树分别如下：



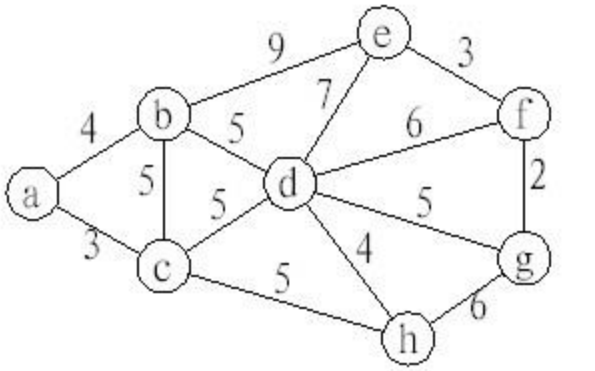


【23， 7，4】（选自《数据结构题集》7.7，必做题）

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；

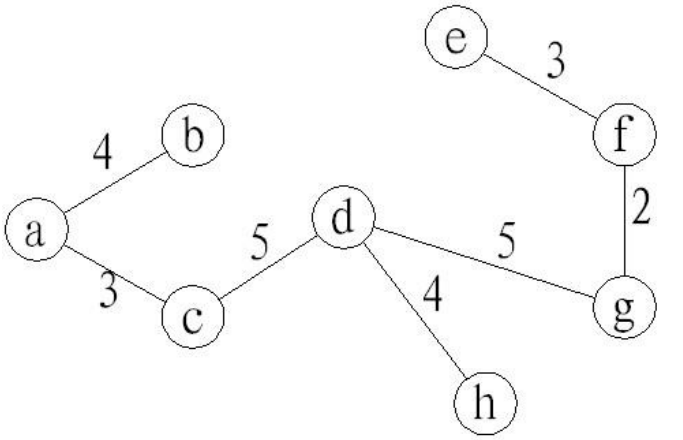
（2） 写出它的邻接表，并按克鲁斯卡尔算法求其最小生成树。



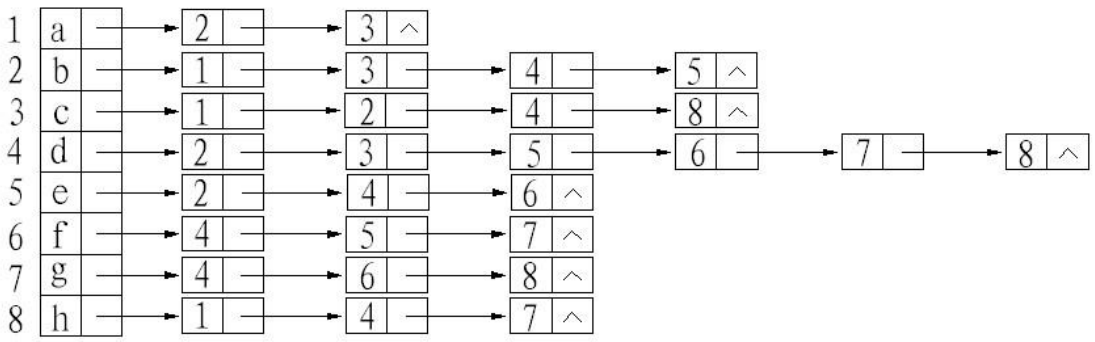
答：(1) 其邻接矩阵为

	a	b	c	d	e	f	g	h
a	0	4	3	0	0	0	0	0
b	4	0	5	5	9	0	0	0
c	3	5	0	5	0	0	0	5
d	0	5	5	0	7	6	5	4
e	0	9	0	7	0	3	0	0
f	0	0	0	6	3	0	2	0
g	0	0	0	5	0	2	0	6
h	0	0	5	4	0	0	6	0

按普里姆算法，得最小生成树为



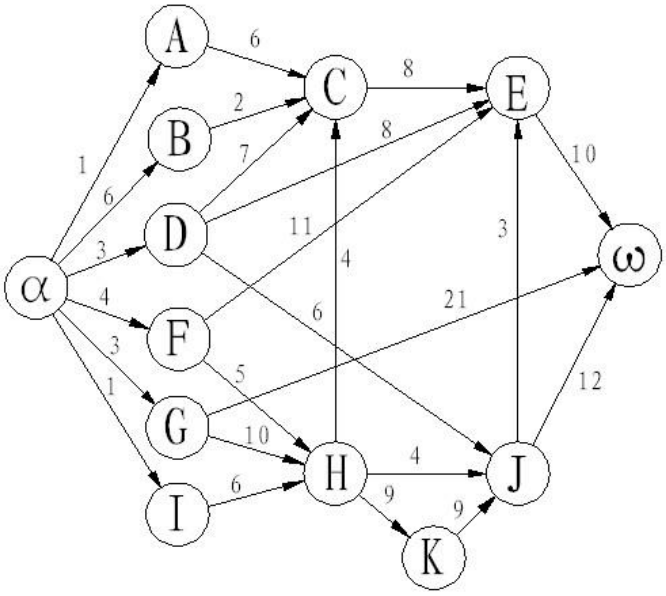
(2) 其邻接表为：



按克鲁斯卡尔算法求得的最小生成树同上。

【24， 7，5】（选自《数据结构题集》7.10，选做题）

对于题 7.10 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



解：e（ai）为活动 ai 的最早开始时间，l（aj）为活动 aj 最迟必须开始的时间，ve（ai）为事件的最早发生时间，vl（aj）为事件的最迟发生时间

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

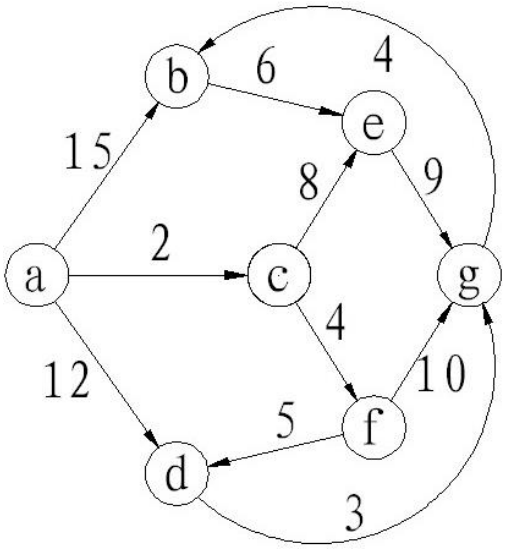
事件	ve（a_i）	vl（a_j）	事件	ve（a_i）	vl（a_j）
α	0	0	G	3	3
A	1	20	H	13	13
B	6	24	I	1	7
C	11	26	J	31	31
D	3	19	K	22	22
E	34	34	ω	44	44

F	4	8			

关键路经：α —G—H—K—J—E— ω

【25， 7，6】（选自《数据结构题集》7.11，必做题）

试利用 Dijkstra 算法求题 7.11 图中从顶点 a 到其他各顶点间的最短路径，写出执行算法中各步的状态。



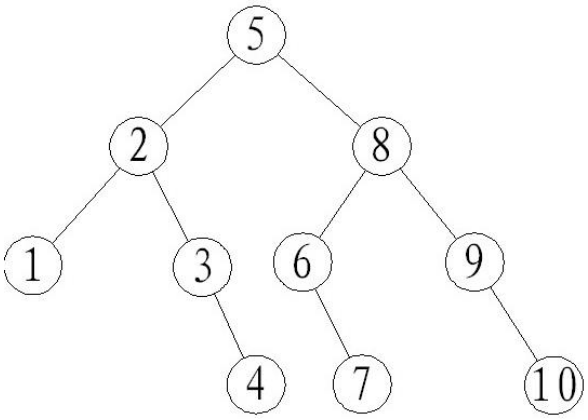
解：

终点 / Dist	b	c	d	e	f	g	S
K＝1	15 (a,b)	2 (a,c)	12 (a,d)				{a,c}
K＝2	15 (a,b)		12 (a,d)	10 (a,c,e)	6 (a,c,f)		{a,c,f}
K＝3	15 (a,b)		11 (a,c,f,d)	10 (a,c,e)		16 (a,c,f,g)	{a,c,f,e}
K＝4	15 (a,b)		11 (a,c,f,d)			16 (a,c,f,g)	{a,c,f,e,d}
K＝5	15 (a,b)					14 (a,c,f,d,g)	{a,c,f,e,d,g}
K＝6	15 (a,b)						{a,c,f,e,d,b}

【26， 9，1】（选自《数据结构题集》9.3，必做题）

画出对长度为 10 的有序表进行折半查找的判定树，并求其等概率时查找成功的平均查找长度。

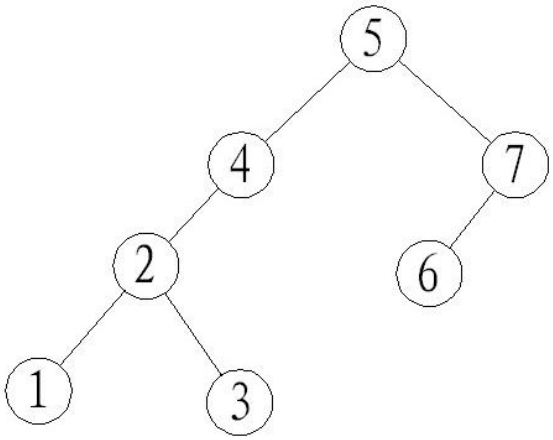
解：其判定树如下：



所以，平均查找长度 ASL＝1×0.1＋2×0.2＋3×0.4＋4×0.3＝2.9

【27， 9，2】（选自《数据结构题集》9.10，选做题）

可以生成如下二叉排序树的关键字的初始排列有几种？请写出其中的任意 5 个。



解：30 种。任写 5 个序列如下：

（5，4，7，6，2，3，1）；
（5，7，4，6，2，3，1）；
（5，4，7，2，3，1，6）；
（5，7，6，4，2，3，1）；
（5，7，6，4，2，1，3）

【28， 5，1】（选自《数据结构题集》9.21，选做题）

在地址空间为 0～16 的散列区中，对以下关键字序列构造两个哈希表：（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。

设哈希函数为 H（x）＝「i/2」，其中 i 为关键字第一个字母在字母表中的序号。

解：（1）用线性探测开放定址法处理冲突

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Apr	Aug	Dec	Feb		Jan	Mar	May	June	July	Sep	Oct	Nov				

ASL_{succ}＝（1×5＋2×3＋4＋5×2＋6）/12=31/12
ASL_{unsucc}＝(5×6/2+9×10/2)/14 = 60/14
（2）用链地址法处理
ASL_{succ}＝（1×7＋2×4＋3）/ 14 = 18 / 12
ASL_{unsucc}＝（1×7＋2×3＋3×3＋4）/ 14 = 26 / 14

【29， 10，4】（选自《数据结构题集》10.1，必做题）
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；（3）快速排序；（4）堆排序
答：（1）直接插入排序
（503），087，512，061，908，170，897，275，653，426
（087，053），512，061，908，170，897，275，653，426
（087，503，512），061，908，170，897，275，653，426
（061，087，503，512），908，170，897，275，653，426
（061，087，503，512，908），170，897，275，653，426
（061，087，170，503，512，908），897，275，653，426
（061，087，170，503，512，897，908），275，653，426
（061，087，170，275，503，512，897，908），653，426
（061，087，170，275，503，512，653，897，908），426
（061，087，170，275，426，503，512，653，897，908）
（3）快速排序
初　始：503，087，512，061，908，170，897，275，653，426
第一趟：426，087，275，061，170，503，897，908，653，512
第二趟：170，087，275，061，426，503，512，653，897，908
第三趟：061，087，170，275，426，503，512，653，897，908
第四趟：061，087，170，275，426，503，512，653，897，908
第五趟：061，087，170，275，426，503，512，653，897，908
（4）堆排序
初　始：（503，087，512，061，908，170，897，275，653，426）
初始堆：（908，653，897，503，426，170，512， 275，061，087）
第一趟：（897，653，512，503，426，170，087， 275，061），908
第二趟：（653，503，512，275，426，170，087， 061），897，908
第三趟：（512，503，170，275，426，061，087），653，897，908
第四趟：（503，426，170，275，087，061），512，653，897，908
第五趟：（426，275，170，061，087），503，512，653，897，908
第六趟：（275，087，170，061），426，503，512，653，897，908
第七趟：（170，087，061），275，426，503，512，653，897，908
第八趟：（087，061），170，275，426，503，512，653，897，908
第九趟：（061），087，170，275，426，503，512，653，897，908

【30， 10，4】（选自《数据结构题集》10.12，必做题）
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）。

解：（1）大顶堆；

（2）否。调整为小顶堆如下：

（12，24，33，65，33，56，48，92，86，70）

（3）大顶堆；

（4）否。调整为小顶堆如下：

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