第一章　绪论

一．选择题

1．数据结构被形式地定义为（K，R），其中 K 是①_B_的有限集合，R 是 K 上的②_D_的有限集合。

①A．算法　　B．数据元素　　C．数据操作　　D．逻辑结构

②A．操作　　B．映象　　C．存储　　D．关系

2．算法分析的目的是①C，算法分析的两个主要方面是②A。

①A．找出数据结构的合理性

B．研究算法中的输入和输出的关系

C．分析算法的效率以求改进

D．分析算法的易懂性和文档性

②A．空间复杂性和时间复杂性

B．正确性和简明性

C．可读性和文档性

D．数据复杂性和程序复杂性

3． 在计算机存储器内表示时，物理地址和逻辑地址相同并且是连续的，称之为(B)

A．逻辑结构　　B．顺序存储结构

C．链表存储结构　　D．以上都不对

4．数据结构中，在逻辑上可以把数据结构分成：（　C　）。

A．动态结构和静态结构　　B．紧凑结构和非紧凑结构

C．线性结构和非线性结构　　D．内部结构和外部结构

5．以下属于顺序存储结构优点的是（　A　）。

A．存储密度大　　B．插入运算方便

C．删除运算方便　　D．可方便地用于各种逻辑结构的存储表示

6．数据结构研究的内容是（　D　）。

A．数据的逻辑结构　　B．数据的存储结构

C．建立在相应逻辑结构和存储结构上的算法　　D．包括以上三个方面

7．链式存储的存储结构所占存储空间（A　）。

A．分两部分，一部分存放结点值，另一部分存放表示结点间关系的指针

B．只有一部分，存放结点值

C．只有一部分，存储表示结点间关系的指针

D．分两部分，一部分存放结点值，另一部分存放结点所占单元数

8．一个正确的算法应该具有 5 个特性，除输入、输出特性外，另外 3 个特性是（ A ）。

A．确定性、可行性、有穷性　　B．易读性、确定性、有效性

C．有穷性、稳定性、确定性　　D．可行性、易读性、有穷性

9．以下关于数据的逻辑结构的叙述中正确的是（　A）。

A．数据的逻辑结构是数据间关系的描述

B．数据的逻辑结构反映了数据在计算机中的存储方式

C．数据的逻辑结构分为顺序结构和链式结构

D．数据的逻辑结构分为静态结构和动态结构

10．算法分析的主要任务是（ C ）。

A．探讨算法的正确性和可读性　　B．探讨数据组织方式的合理性

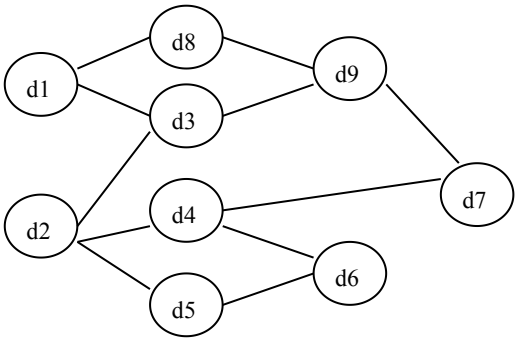
C．为给定问题寻找一种性能良好的解决方案　　D．研究数据之间的逻辑关系

二．解答

设有一数据的逻辑结构为：B=(D, S)，其中：

D={d1, d2, …, d9}

S={<d1,d3>, <d1, d8>, <d2, d3>, <d2, d4>, <d2, d5>, <d3, d9>, <d4, d7>, <d4, d6>, <d5, d6>, <d8, d9>, <d9, d7> }画出这个逻辑结构示意图。



第二章 线性表

一、选择题

1．下述哪一条是顺序存储结构的优点？（ A）
A．存储密度大 B．插入运算方便 C．删除运算方便 D．可方便地用于各种逻辑结构的存储表示
2．下面关于线性表的叙述中，错误的是哪一个？（ B）
A．线性表采用顺序存储，必须占用一片连续的存储单元。
B．线性表采用顺序存储，便于进行插入和删除操作。
C．线性表采用链接存储，不必占用一片连续的存储单元。
D．线性表采用链接存储，便于插入和删除操作。
3．若某线性表最常用的操作是存取任一指定序号的元素和在最后进行插入和删除运算，则利用（A ）存储方式最节省时间。
A．顺序表 B．双链表 C．带头结点的双循环链表 D．单循环链表
4．某线性表中最常用的操作是在最后一个元素之后插入一个元素和删除第一个元素，则采用（ D）存储方式最节省运算时间。
A．单链表 B．仅有头指针的单循环链表 C．双链表 D．仅有尾指针的单循环链表
5．在一个长度为 n 的顺序表中删除第 i 个元素(0<=i<=n)时，需向前移动（A）个元素
A．n-i B．n-i+l C．n-i-1 D．i
6．从一个具有 n 个结点的单链表中查找其值等于 x 的结点时，在查找成功的情况下，需平均比较（C）个元素结点
A．n/2 B．n C．（n+1）/2 D．（n-1）/2
7．设单链表中指针 p 指向结点 m，若要删除 m 之后的结点（若存在），则需修改指针的操作为（A）
A．p->next=p->next->next; B．p=p->next;
C．p=p->next->next; D．p->next=p;
8．在一个单链表中，已知 q 结点是 p 结点的前趋结点，若在 q 和 p 之间插入 s 结点，则须执行（B）
A．s->next=p->next; p->next=s
B．q->next=s; s->next=p
C．p->next=s->next; s->next=p
D．p->next=s; s->next=q
9．线性表的顺序存储结构是一种（A）的存储结构。
A．随机存取 B．顺序存取 C．索引存取 D．散列存取

二、填空

1．在线性表的顺序存储中，元素之间的逻辑关系是通过__物理位置相邻____决定的；在线性表的链接存储中，元素之间的逻辑关系是通过__指针____决定的。
2．在双向链表中，每个结点含有两个指针域，一个指向___.直接前驱____结点，另一个指向___直接后继____结点。
3．当对一个线性表经常进行存取操作，而很少进行插入和删除操作时，则采用_顺序_____存储结构为宜。相反，当经常进行的是插入和删除操作时，则采用___链式_____存储结构为宜。

三、算法设计

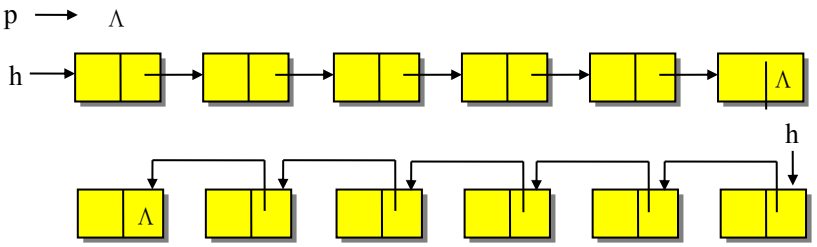
1．设有一个正整数序列组成的有序单链表（按递增次序有序，且允许有相等的整数存在），试编写能实现下列功能的算法（要求用最少的时间和最小的空间）

①确定在序列中比正整数 x 大的数有几个（相同的数只计算一次）

②将单链表中比正整数 x 小的偶数从单链表中删除

```
int count(Linklist h,int x)
{
   int num=0;
   Linknode *p;
   p=h->next;
   while(p&&p->data<=x)//p 指针向后移动，直至 p 指向第一个值大于 x 的结点
      p=p->next;
   while(p)
      if(p->next&&p->data==p->next->data)
      //若 p 没有指向链表中同一数值的最后一个结点，则向后移动
         p=p->next;
      else
//若 p 指向数值相同的结点中的最后一个，则 num 加 1，p 指针后移，继续执行 while 循环
      {
         num++;
         p=p->next;
      }
    return num;
}
②
void delevenl(Linklist &h,int x)
{
   Linknode *p,*r;
   p=h->next;r=h;
   while(p&&p->data<x)
   {
      if(p->data%2==0)
      {
         r->next=p->next;
         free(p);
         p=r->next;
      }
      else
      {
         r=p;
         p=p->next;
      }
   }
}
```

2．设有一个表头指针为 h 的单链表。试设计一个算法，通过遍历一趟链表，将链表中所有结点的链接方向逆转，如下图所示。要求逆转结果链表的表头指针 h 指向原链表的最后一个结点。




```
2．void converse(Linklist &h)
{
    Linknode *p,*q;
    p=h->next;
    h->next=NULL;
    q=p->next;
    while(q)
    {
        p->next=h;
        h=p;
        p=q;
        q=q->next;
    }
    p->next=h;
    h=p;
}
```

3．设计算法将一个带头结点的单链表 A 分解为两个具有相同结构的链表 B、C，其中 B 表的结点为 A 表中值小于零的结点，而 C 表的结点为 A 表中值大于零的结点（链表 A 的元素类型为整型，要求 B、C 表利用 A 表的结点）。

```
3．void decompose(Linklist La,Linklist &Lb,Linklist &Lc)
{
    Linknode *p;
    Lc=(Linknode *)malloc(sizeof(Linknode));
    Lc->next=NULL;
    p=La->next;
    Lb=La;
    Lb->next=NULL;
    while(p)
    {
        La=p->next;
        if(p->data>0)
        {
            p->next=Lc->next;
            Lc->next=p;
```

```
            }
            else
            {
            p->next=Lb->next;
                  Lb->next=p;
            }
            p=La;
      }
}
```

4. 假设链表 A、B 分别表示一个集合，试设计算法以判断集合 A 是否是集合 B 的子集，若是，则返回 1，否则返回 0，并分析算法的时间复杂度。

```
4．int subset(LinkList la, LinkList lb)
{   LinkNode * pa,*pb;
   pa=la->next;
   while(pa)
   {   pb=lb->next;
       while(pb&&(pb->data!=pa->data)) pb=pb->next;
       if(!pb) return 0;
       pa=pa->next;
      }
   return 1;
}
```

算法时间复杂度 O（A.Length*B.Length）

5．设有一单循环链表 la，其结点有三个域：prior、data 与 next，其中 data 为数据域,，next 域指向直接后继，prior 域应指向直接前驱，但目前空着。试写一算法将此单循环链表改造为双向循环链表。

```
5．void priorset(DuLinkList &la)
{ p=la;q=la->next;
  while(q!=la){q->prior=p; p=q;q=q->next;}
   q->prior=p;
}
```

第三章 栈和队列

一、选择题

1．已知栈的最大容量为 4。若进栈序列为 1，2，3，4，5，6，且进栈和出栈可以穿插进行,则可能出现的出栈序列为（ C）

A.5，4，3，2，1，6　　B.2，3，5，6，1，4

C.3，2，5，4，1，6　　D.1，4，6，5，2，3

设有一个栈，元素的进栈次序为 A, B, C, D, E,下列是不可能的出栈序列（C ）

A．A, B, C, D, E　　B．B, C, D, E, A

C．E, A, B, C, D　　D．E, D, C, B, A

2．在一个具有 n 个单元的顺序栈中，假定以地址低端（即 0 单元）作为栈底，

以 top 作为栈顶指针，当做出栈处理时，top 变化为（C ）

A．top 不变　　B．top=0　　C．top--　　D．top++

3．向一个栈顶指针为 hs 的链栈中插入一个 s 结点时，应执行（B ）

A．hs->next=s;

B．s->next=hs;　hs=s;

C．s->next=hs->next;hs->next=s;

D．s->next=hs; hs=hs->next;

4．在具有 n 个单元的顺序存储的循环队列中，假定 front 和 rear 分别为队头指针和队尾指针，则判断队满的条件为（ D）

A．rear%n= = front　　B．(front+1) %n= = rear

C．rear%n -1= = front　D．(rear+1)%n= = front

5．在具有 n 个单元的顺序存储的循环队列中，假定 front 和 rear 分别为队头指针和队尾指针，则判断队空的条件为（ C）

A．rear%n= = front　　B．front+1= rear

C．rear= = front　　D．(rear+1)%n= front

6．在一个链队列中，假定 front 和 rear 分别为队首和队尾指针，则删除一个结点的操作为（ A）

A．front=front->next　　B．rear=rear->next

C．rear=front->next　　D．front=rear->next

7．某堆栈的输入序列为 1，2，3，…，n，输出序列的第一个元素是 n，则第 i 个输出元素为（ C）

A．i　　B．n-i　　C．n-i+1　D．哪个元素无所谓

8．用不带头结点的单链表存储队列时,其队头指针指向队头结点,其队尾指针指向队尾结点，则在进行删除操作时（　D　）。

A．仅修改队头指针　　B．仅修改队尾指针

C．队头、队尾指针都要修改　D．队头,队尾指针都可能要修改

二、解答题

1．一个双向栈 S 是在同一向量空间内实现的两个栈，它们的栈底分别设在向量空间的两端。 试为此双向栈设计初始化 InitStack (S) 、入栈 Push(S , i , x) 和出栈 Pop(S , i)等算法， 其中 i 为 0 或 1， 用以表示栈号。

1．双向栈


```
//双向栈类型定义
#define STACK_SIZE 100;
Typedef struct {
        SElemType    * base_one, * base_two;//栈底指针
```

```
        SElemType    * top_one, * top_two;//栈顶指针
        int stacksize;
            } SqStack;

Status InitStack ( SqStack &S) {
//初始化双向栈
    S.base_one=S.top_one=( SElemType *)malloc(STACK_SIZE * sizeof(SElemType));// 第一个栈底和栈顶指向数组起点
    S.base_two=S.top_two=S.base_one +STACK_SIZE-1;// 第二个栈底和栈顶指向数组终点
    S.stacksize= STACK_SIZE ;
    return OK;
    }//InitStack

Status Push ( SqStack &S, int i, SElemType e){
//入栈操作，i=0 时将 e 存入前栈，i=1 时将 e 存入后栈
    if( S.top_two < S.top_one)     return OVERFLOW;//栈满，不考虑追加空间
    if( i = = 0 )
    *S.top_one++ = e;
    else
    *S.top_two-- = e;
    return OK;
    }//Push

SElemType    Pop ( SqStack &S, int i){
//出栈操作，i=0 出前栈，i=1 出后栈
if( i==0 ) {
        if ( S.top_one==S.base_one) return ERROR;
        S.top_one--; e=*S.top_one;
          }
else {
       if ( S.top_two==S.base_two) return ERROR;
        S.top_two++; e=*S.top_two;
          }
return e;
}//Pop
```

2．利用两个栈 S1、S2 模拟一个队列时，如何使用栈的运输实现队列的插入、删除运算。

```
2．#define M 3
struct Stack{
    Qelemtype data[M];
    int top;
};
struct Queue{
```

```
    Stack s1;
    Stack s2;
};
void InitQueue(Queue &Q)//初始化队列
{
    Q.s1.top=0;
    Q.s2.top=0;
}
int IsEmpty(Queue &Q)//判断队列是否为空
{
    if(Q.s1.top==0&&Q.s2.top==0)
        return 1;
    if(Q.s2.top==0&&Q.s1.top!=0)
    {
        while(Q.s1.top!=0)
            Q.s2.data[Q.s2.top++]=Q.s1.data[--Q.s1.top];
    }
    return 0;
}

int IsFull(Queue &Q)
{
    if(Q.s1.top==M&&Q.s2.top!=0)
        return 1;
    if(Q.s1.top==M&&Q.s2.top==0)
    {
        while(Q.s1.top!=0)
            Q.s2.data[Q.s2.top++]=Q.s1.data[--Q.s1.top];
        return 0;
    }
    if(Q.s1.top!=M)
        return 0;
}

void InQueue(Queue &Q,Qelemtype e)
{
    if(IsFull(Q))
    {
        cout<<"OVERFLOW"<<endl;
        return;
    }
    Q.s1.data[Q.s1.top++]=e;
}
```

```
void DeQueue(Queue &Q,Qelemtype &e)
{
    if(IsEmpty(Q))
    {
        cout<<"UNDERFLOW"<<endl;
        return;
    }
    e=Q.s2.data[--Q.s2.top];
}
```

3．已知一个栈 S 的输入序列为 abcd，下面两个序列能否通过栈的 push 和 pop 操作输出？如果能，请写出操作序列；如果不能，请写明原因。

(1)dbca　　(2)cbda

3．(1)不能　（2）可以 原因略

4．假设以带头结点的循环链表表示队列，并且只设一个指针指向队尾元素结点(注意不设头指针) ，试编写相应的置空队、判队空 、入队和出队等算法。

4．
```
struct QueueNode
{
    Qelemtype data;
    QueueNode *next;
};

struct LinkQueue
{
    QueueNode *rear;
};

void InitQueue(LinkQueue &Q)   //置空队
{
    Q.rear=new QueueNode;
    Q.rear->next=Q.rear;
}

int EmptyQueue(LinkQueue Q)   //判队空
{
    return Q.rear==Q.rear->next;
}
void EnQueue(LinkQueue &Q, Qelemtype e)//元素入队
{
    QueueNode *p;
//新建一个结点，并置其值为 e
    p=new QueueNode;
    p->data=e;
```

```
//将结点插入队列末尾
      p->next=Q.rear->next;
      Q.rear->next=p;
//修改队尾指针
      Q.rear=p;
}
void DeQueue(LinkQueue &Q,Qelemtype &e)   //元素出队
{
      QueueNode *p;
      if (EmptyQueue(Q))   //若队中无元素，则无法执行出队操作
      {
            cout<<"error"<<endl;
            return;
      }
      p=Q.rear->next->next;   //让 p 指向队头元素
      e=p->data;
   if(p==Q.rear)//如队中只有一个元素，则要 rear 指向头结点，头结点的 next 指针指向自身
   {
            Q.rear=Q.rear->next;
        Q.rear->next=Q.rear;
    }
       else       //若队中不只一个元素，则删除 p 所指向的结点。
                  Q.rear->next->next=p->next;
      delete p;//释放被删除结点的存储空间
}
```

5．假设以 I 和 O 表示入栈和出栈操作，栈的初态和终态均为空，入栈和出栈的操作序列可表示为仅由 I 和 O 组成的序列。称可以操作的序列为合法序列，否则称为非法序列。

(1)下面所示的序列中哪些是合法的？

A．IOIIOIOO　　B.IOOIOIIO　　C.IIIOIOIO　　D.IIIOOIOO

(2)通过对问题⑴的分析，写出一个算法判定所给的操作序列是否合法。若合法返回 TRUE，否则返回 FALSE。（假定被判定的操作序列已经存入一维数组中）

```
5．typedef struct
{
 char data[10];
 int top;
}stack;

int descion(char *str)
{
 stack s;
 int i;
```

```
 s.top=0;
 for(i=0;str[i];i++)
 {
  switch(str[i])
  {
     case 'I':s.data[s.top++]=str[i];break;//若为 I，则如栈
     case 'O':if(s.top==0) return 0;s.top--;//若为 O，则从栈中弹出一个 I
   }
  }
 if(s.top==0) return 1;
 else return 0;
}
```

第四章　串

一．选择题

1. 若串 S='software',其子串的数目是（　B）

A.8　　B.37　　C.36　　D.9

2. 设有两个串 p 和 q,求 q 在 p 中首次出现的位置的运算称作（B　）

A.连接　　B.模式匹配　　C.求串长　　D.求子串

3．设字符串 S1=“ABCDEFG”，S2=“PQRST”，则运算：

S=CONCAT（SUBSTR（S1，2，LEN（S2））；SUBSTR（S1，LEN（S2），2））；后的串值为 （　D）

A．A BCDEF　　B．BCDEFG　　C．BCDPQRST　　D．BCDEFEF

4. 下面的说法中，只有（A　）是正确的

A.串是一种特殊的线性表　　B. 串的长度必须大于零

C.串中元素只能是字母　　D. 空串就是空白串

5. 两个字符串相等的条件是（D　）

A.两串的长度相等

B.两串包含的字符相同

C.两串的长度相等，并且两串包含的字符相同

D.两串的长度相等，并且对应位置上的字符相同

二、填空题

1. 令 t1=“aaab”, t2=“abcabaa”, t3=“abcaabbabcabaacba”，试分别求出他们的 nex 函数值 __0123__ ，__0111232__ ，____01112231234532211___。

2.空格串的长度为__. 空格数___ ，空串的长度为___0___ 。

3.设串 S=’How are you’，则串的长度为____11____ 。

三、问答题

回文是指正读反读均相同的字符序列，如"abba"和"abdba"均是回文，但"good"不是回文。试写一个算法判定给定的字符变量是否为回文。

```
Status IsHuiwen( char *S)
{ i=0;
  while(s[i]!=’ \0’ ) i++;
i=i-1; j=0;
while(j<i)
   if(s[j]!=s[i]) return 0;
   else {j++;i--;}
return 1;
}
```

第五章 数组和广义表

一．选择题

1. 设有数组 A[i,j]，数组的每个元素长度为 3 字节，i 的值为 1 到 8 ，j 的值为 1 到 10，数组从内存首地址 BA 开始顺序存放，当用以列为主存放时，元素 A[5，8]的存储首地址为(B)

A. BA+141　　B. BA+180　　C. BA+222　　D. BA+225

2. 假设以行序为主序存储二维数组 A=array[1..100，1..100]，设每个数据元素占 2 个存储单元，基地址为 10，则 LOC[5，5]=（ B ）

A. 808　　B. 818　　C. 1010　　D. 1020

3 对稀疏矩阵进行压缩存储目的是（ C ）

A．便于进行矩阵运算　B．便于输入和输出　C．节省存储空间　D．降低运算的时间复杂度

4. 广义表 A=(a,b,(c,d),(e,(f,g))),则下面式子的值为（ D ）。

Head(Tail(Head(Tail(Tail(A)))))

A. (g)　　B. (d)　　C. c　　D. d

5. 设广义表 L=((a,b,c))，则 L 的长度和深度分别为（ B ）。

A. 1 和 1　　B. 1 和 3　　C. 1 和 2　　D. 2 和 3

6. 假设以三元组表表示稀疏矩阵，则与如图所示三元组表对应的 4×5 的稀疏矩阵是（注：矩阵的行列下标均从 1 开始）（ B ）

A. $\begin{pmatrix} 0 & -8 & 0 & 6 & 0 \\ 7 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -5 & 0 & 4 & 0 & 0 \end{pmatrix}$

B. $\begin{pmatrix} 0 & -8 & 0 & 6 & 0 \\ 7 & 0 & 0 & 0 & 3 \\ -5 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$

0	1	2	-8
1	1	4	6
2	2	1	7
3	2	5	3
4	3	1	-5
5	3	3	4

题 6 图

C. $\begin{pmatrix} 0 & -8 & 0 & 6 & 0 \\ 0 & 0 & 0 & 0 & 3 \\ 7 & 0 & 0 & 0 & 0 \\ -5 & 0 & 4 & 0 & 0 \end{pmatrix}$

D. $\begin{pmatrix} 0 & -8 & 0 & 6 & 0 \\ 7 & 0 & 0 & 0 & 0 \\ -5 & 0 & 4 & 0 & 3 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$

7. 一个非空的广义表的表尾（ B）

A.不能是子表　　B.只能是子表　　C.只能是原子　　D.是原子或子表

8. 如果将矩阵 An×n 的每一列看成一个子表，整个矩阵看成是一个广义表 L，即 L=((a11,a21,…,an1),(a12,a22,…,an2),…,（a1n,a2n,…,ann）),并且可以通过求表头 head 和求表尾 tail 的运算求取矩阵中的每一个元素，则求得 a21 的运算是（ A ）

A. head (tail (head (L)))　　B. head (head(head(L)))

C. tail (head (tail (L)))　　D. head (head (tail (L)))

二．填空题

1.已知三对角矩阵 A[1..9,1..9]的每个元素占 2 个单元，现将其三条对角线上的元素逐行存储在起始地址为 1000 的连续的内存单元中，则元素 A[7,8]的地址为__1038____

2. 设广义表 L=((),()), 则 head(L)是_()；tail(L)是(())_；L 的长度是 2_；深度是 2_。

3. 广义表 A=（（（a，b），（c，d，e））），试用求表头和表尾的操作 Head( )和 Tail( )取出 A 中的原子 e 的操作是: _ Head(Tail(Tail(Head(Tail(Head(A))))))__.

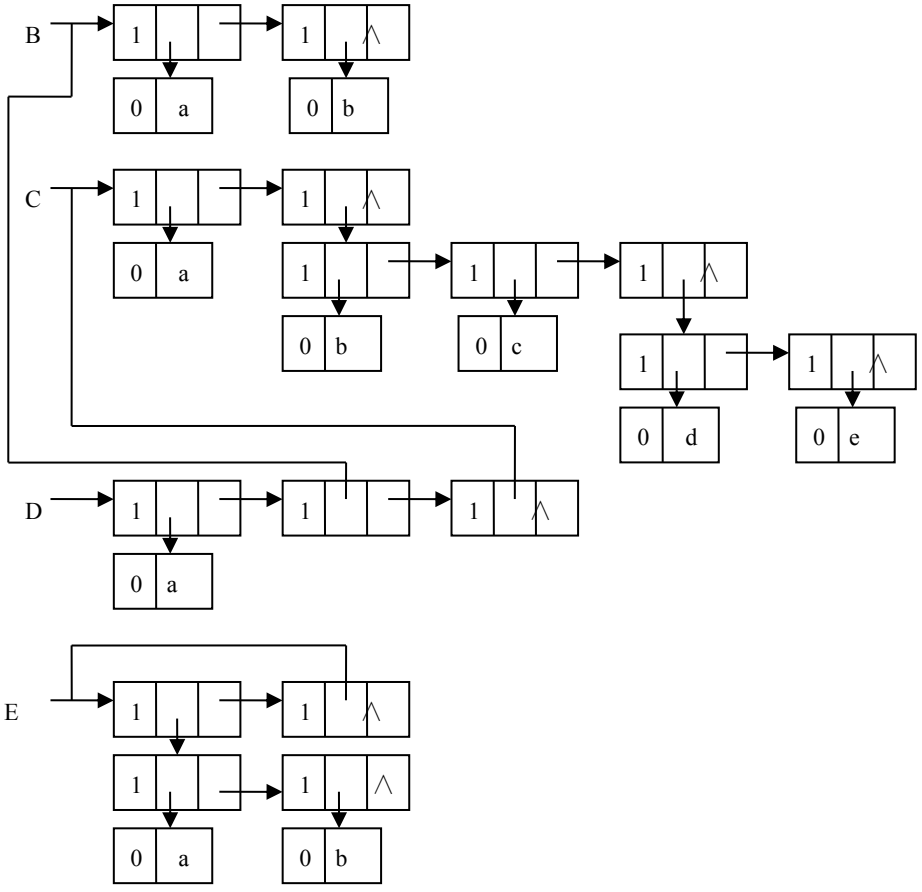
三．解答下列各题

1．已知一个 6 行 5 列的稀疏矩阵中非零元的值分别为：-90，41，-76，28，-54，65，-8，它们在矩阵中的列号依次为：1，4，5，1，2，4，5。当以带行表的三

元组表作存储结构时，其行表中的值依次为 0，0，2，2，3，5（行列下标均从 1 开始），写出该稀疏矩阵。

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ -90 & 0 & 0 & 41 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -76 \\ 28 & -54 & 0 & 0 & 0 \\ 0 & 0 & 0 & 65 & -8 \end{pmatrix}$$

2．写出广义表 B=(a,b),C=(a,(b,c,(d,e))),D=(a,B,C),E=((a,b),E)的存储结构（任一种存储方法均可）



第六章 树和二叉树

一．选择题{CDDAC ADDED BCCBB CACCC}

1．如果在数据结构中每个数据元素只可能有一个直接前驱，但可以有多个直接后继，则该结构是（ ）

A. 栈 B. 队列 C. 树 D. 图

2．设树 T 的度为 4，其中度为 1，2，3 和 4 的结点个数分别为 4，2，1，1 则 T 中的叶子

数为（ ）
A．5　　B．6　　C．7　　D．8
3．已知一棵含50个结点的二叉树中只有一个叶子结点，则该树中度为1的结点个数为（ ）
A．0　　B．1　　C．48　　D．49
4．树的先根序列等同于与该树对应的二叉树的（ ）
A．先序序列　　B．中序序列　C．后序序列　　D．层序序列
5．用二叉链表表示具有n个结点的二叉树时，值为空的指针域的个数为（ ）
A．n-1　　B．n　　C．n+1　　D．2n
6．设森林F对应的二叉树为B，它有m个结点，B的根为p,p的右子树结点个数为n,森林F中第一棵树的结点个数是（ ）
A．m-n　　B．m-n-1　　C．n+1　　D．条件不足，无法确定
7．设树T的度为4，其中度为1，2，3和4的结点个数分别为4，2，1，1，则T中的叶子数为（ ）
A．5　　B．6　　C．7　　D．8
8．设森林F中有三棵树，第一，第二，第三棵树的结点个数分别为M1，M2和M3。与森林F对应的二叉树根结点的右子树上的结点个数是（ ）
A．M1　　B．M1+M2　　C．M3　　D．M2+M3
9．一棵完全二叉树上有1001个结点，其中叶子结点的个数是（ ）
A．250　　B．500　　C．254　　D．505　　E．以上答案都不对
10．有n个叶子的哈夫曼树的结点总数为（ ）
A．不确定　　B．2n　　C．2n+1　　D．2n-1
11．一棵二叉树高度为h,所有结点的度或为0，或为2，则这棵二叉树最少有（ ）结点
A．2h　　B．2h-1　　C．2h+1　　D．h+1
12．将有关二叉树的概念推广到三叉树，则一棵有244个结点的完全三叉树的高度（ ）
A．4　　B．5　　C．6　　D．7
13．若度为m的哈夫曼树中，其叶结点个数为n，则非叶结点的个数为（ ）
A.n-1　　B.⌊n/m⌋-1　　C.⌈(n-1)/(m-1)⌉　　D.⌈n/(m-1)⌉-1　　E.⌈(n+1)/(m+1)⌉-1
14．若下面几个符号串编码集合中，不是前缀编码的是（ ）。
A．{0,10,110,1111}　　B．{11,10,001,101,0001}
C．{00,010,0110,1000}　　D．{b,c,aa,ac,aba,abb,abc}
15．一棵二叉树的前序遍历序列为ABCDEFG，它的中序遍历序列可能是（ ）
A．CABDEFG　　B．ABCDEFG　　C．DACEFBG　　D．ADCFEG
16．线索二叉树是一种（ ）结构。
A．逻辑　　B．逻辑和存储　　C．物理　　D．线性
17．引入二叉线索树的目的是（ ）
A．加快查找结点的前驱或后继的速度　　B．为了能在二叉树中方便的进行插入与删除
C．为了能方便的找到双亲　　D．使二叉树的遍历结果唯一
18．n个结点的线索二叉树上含有的线索数为（ ）
A．2n　　B．n－1　　C．n＋1　　D．n
19．若X是二叉中序线索树中一个有左孩子的结点，且X不为根，则x的前驱为（ ）
A.X的双亲　　B.X的右子树中最左的结点
C.X的左子树中最右结点　　D.X的左子树中最右叶结点
20．某二叉树的前序序列和后序序列正好相反，则该二叉树一定是（ ）的二叉树
A．空或只有一个结点　　B．任一结点无左子树

C．高度等于其结点数　　D．任一结点无右子树

二．填空题

1. 假定一棵树的嵌套括号表示法为 A （ B （ E ）， C （ F （ H ， I ， J ）， G ），D ），则该树的度为_____，树的深度为_____，终端点的个数为____，但分支结点的个数为_____，双分支结点为_____，三分支结点的个数为_____， C 结点的双亲结点为_____，其孩子结点为_____和为_____结点。

2． 一棵深度为 K 的满二叉树结点总数为_____ ，一棵深度为 K 的完全二叉树的结点总数的最小值为_____，最大值为_____。

3． 由三个结点构成的二叉树，共有_____种不同的形态。

4． 具有 100 个叶子结点的完全二叉树的深度为____

5． 高为 h（h>0）的满二叉树对应森林的由____棵树构成。

三．已知一个二叉树的中序序列为 CBEDAHGIJF,后序序列为 CEDBHJIGFA。

1．画出该二叉树。

2．画出该二叉树的先序线索二叉树。

四．试找出分别满足下列条件的所有二叉树：

1．先序序列和中序序列相同。

2．中序序列和后序序列相同。

3．先序序列和后序序列相同。

五、设二叉树用二叉链表表示，设计算法求二叉树的高度。

```
int Depth(BiTree t)（后序遍历）
{    if(！t)
        d=0;
     else
    {dl=Depth(t->lchild);
     dr=Depth(t->rchild);
     d=1+(dl>dr?dl:dr); }
    return d;
}
```

六、设 T 是一棵具有 n 个结点的二叉树，若给定二叉树 T 的先序序列和中序序列，并假设 T 的先序序列和中序序列分别放在数组 PreOrder[1..n]和 InIrder[1..n]中，设计一个构造二叉树 T 的二叉链表存储结构的算法。

六 //根据二叉树的先序序列和中序序列创建二叉链表

```
void createBiTree(char pre[],char in[],int start,int end,int &count,BiTree &T)
//按先序次序创建二叉链表，pre 为存放先序序列的字符数组，in 为存放中序序列的字符数组，
//count 为二叉树的根结点在先序序列中的序号
//start 和 end 为以 pre[count]为根的二叉树在中序序列中的起始位置和结束位置
//T 为二叉链表的根指针，
{
    if(start > end) T=0;
    else
```

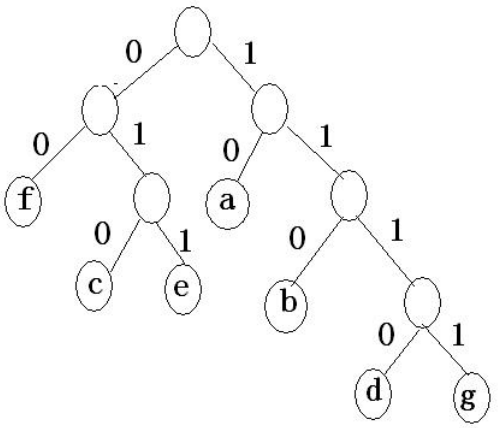
```
{   T=(BiTNode*) malloc(sizeof(BiTNode)); //新建根结点
    if(!T) exit(0);
    T->data=pre[count];T->lchild=T->rchild=0;

     for(int i=0;in[i]!='\0';i++) //查找根结点在中序序列 in[ ]中的下标
            if(in[i]==pre[count]) break;
     count++;
     if(in[i]=='\0') cout<<"input error"<<endl;
     else {
createBiTree(pre,in,start,i-1,count,T->lchild);//根据先序序列和中序序列创建左子树
createBiTree(pre,in,i+1,end,count,T->rchild); //根据先序序列和中序序列创建右子树
 }
```

七、设用于通信的电文由字符集｛a,b,c,d,e,f,g｝中的字母构成，它们在电文中出现的频率分别为｛0.31,0.16,0.10,0.08,0.11,0.20,0.04｝，回答下列问题：

⑴为这 7 个字母设计哈夫曼编码

⑵若对这 7 个字母进行等长编码，至少需要几位二进制数？



、 哈夫曼树

a: 10　b: 110　c: 010　d: 1110　e: 011　f:: 00　g:1111

等长编码至少需要 3 位二进制位。

八．设计算法以输出二叉树中先序序列的前 k（k>0）个结点的值。

```
八 void    preintraverse(BiTree T, int &n)
{   if(T)
     { n++;
     if(n<=k) {cout<<T->data;
              intraverse(T->lchild,n);
              intraverse(T->rchild ,n);}
     }
}
```

九．编写算法，对一棵二叉树统计叶子的个数

```
九、void CountLeaf(BiTree t，int &count)（先序遍历）
{    if(t)
   { if((t->lchild==NULL)&&(t->rchild==NULL))
            count++;
```

```
    CountLeaf(t->lchild，&count);
    CountLeaf(t->rchild,&count);
}
```

第七章 图

一．选择题

1．n 个顶点，e 条边的有向图的邻接矩阵中非零元素有__C____个。

A.n　　B.2e　　C.e　　D.n+e

2．用邻接表存储图所用的空间大小（A）

A．与图的顶点数和边数都有关

B．只与图的边数有关

C．只与图的顶点数有关

D．与边数的平方有关

3．有 n 条边的无向图的邻接表存储法中，链边中结点的个数是（B ）个。

A．n　　B．2n　　C．n/2　　D．n*n

4．一个带权无向连通图的最小生成树（A ）。

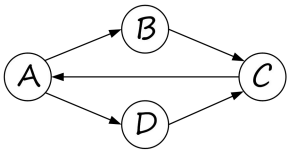
A．有一棵或多棵 .　　B．只有一棵　　C．一定有多棵　　D．可能不存在

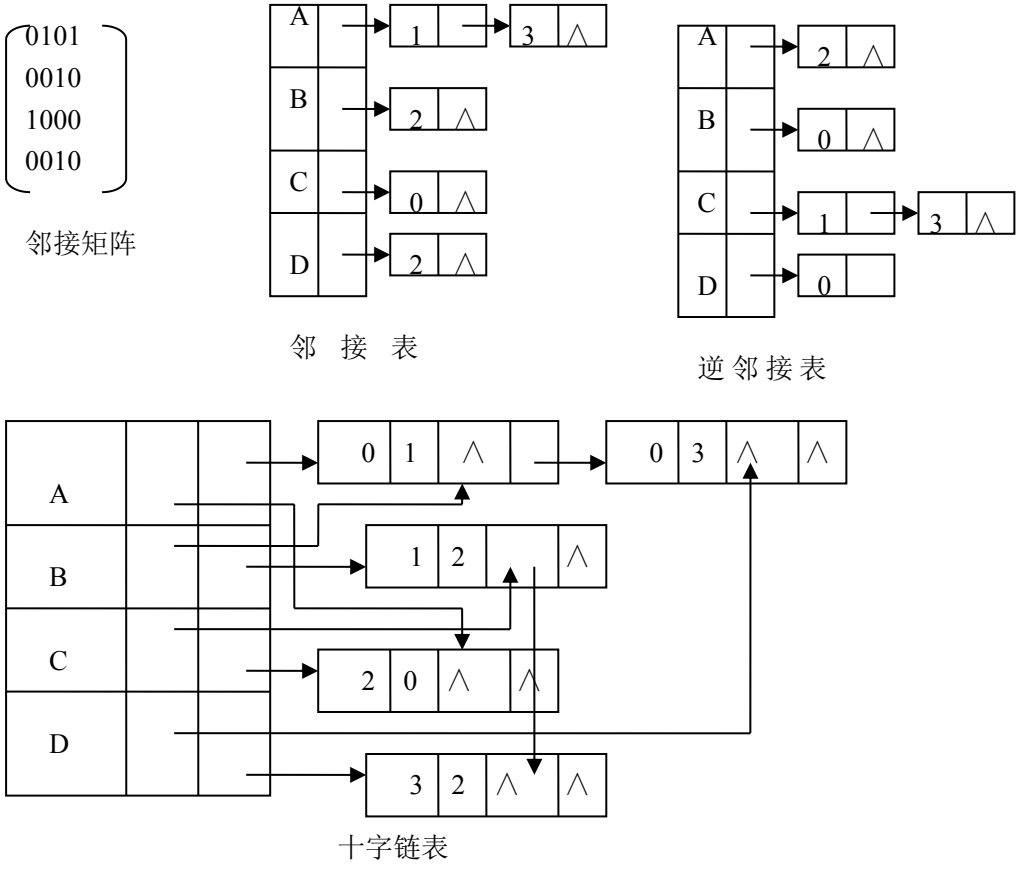
二．如下所示有向图：

1．请给出每个顶点的度，入度和出度。

	A	B	C	D
入度	1	1	2	1
出度	2	1	1	1
度	3	2	3	2

2．请画出其邻接矩阵、邻接表、逆邻接表、十字链表。



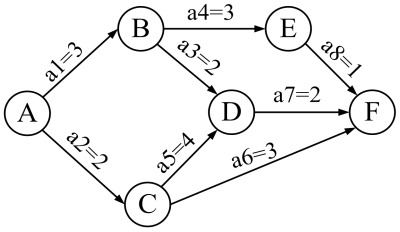


三．试对下图所示的 AOE 网络，解答下列问题。

1．求每个事件的最早发生时间 ve [i]和最迟发生时间 vl[i]。

2．求每个活动的最早开始时间 ee(s)和最迟开始时间 el(s)。

3．指出哪些活动加速可使整个工程提前完成。

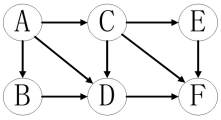


事件	A	B	C	D	E	F
最早发生时间 ve[i]	0	3	2	6	6	8
最迟发生时间 vl[i]	0	4	2	6	7	8

活动	a1	a2	a3	a4	a5	a6	a7	a8
最早开始时间	0	0	3	3	2	2	6	6
最迟开始时间	1	0	4	4	2	5	6	7

关键活动为 a2　a5　a7,加速这些关键活动可使整个工程提前完成。

四．写出下图所示的 AOV 网的所有拓扑有序序列。



四、拓扑有序序列

ABCDEF

ABCEDF

ACBDEF

ACBEDF

ACEBDF

第九章　查找

一．填空题

１．采用二分法进行查找的查找表，应选择___顺序_____方式的存储结构

２．设在有序表 A[0……9]中进行二分查找，比较一次查找成功的结点数为_1____，比较二次查找成功的结点数为__2____，比较三次查找成功的结点数为__4___，比较四次查找成功的结点数为__3___，比较五次查找成功的结点数为__0___，平均查找长度为___（1+2*2+3*4+4*3）/10=2.9___。

（3）设在线性表 R[0……59]中进行分块查找，共分 10 块，每块长度为 6，若利用顺序查找法对索引表和子块进行查找，则查找每个元素的平均查找长度为___9_____。

二．选择题

１．对线性表进行二分查找时，要求线性表必须（　B　）

A．键值有序的链接表　　B．键值有序的顺序表

C．链接表但键值不一定有序　　D．顺序但键值不一定有序

２．有一个有序表{1，4，6，10，18，35，42，53，67，71，78，84，92，99},当用二分查找法查找键值为 84 的结点时，经（　C　）比较后查找成功。

A．2　　B．　3　　C．4　　D．12

３．顺序检索一个具有 n 个数据元素的线性表，其时间复杂度为_____，二分检索一个具有 n 个数据元素的线性表，其时间复杂度为（　AB　）

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

４．设散列表长度为 m，散列函数为 H（key）=key%p，为了减少发生冲突的可能性，p 应取（　B　）

A．小于 m 的最大奇数　　B．小于 m 的最大素数

C．小于 m 的最大偶数　　D．小于 m 的最大合数

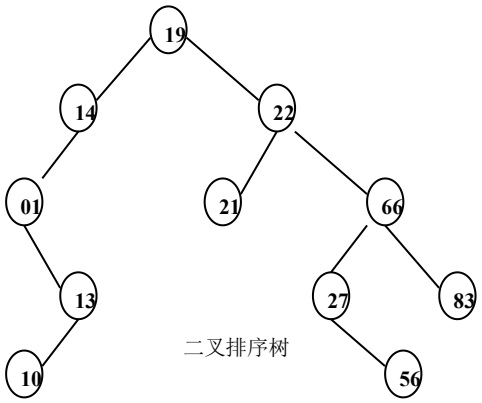
三．解答题

１．给定表{19,14,22,01,66,21,83,27,56,13,10}

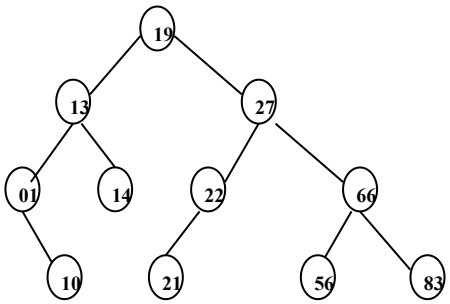
①试按元素在表中的顺序构造一棵二叉排序树；

②判断该二叉排序树是否平衡，若不平衡，调整其为平衡二叉树。

1、二叉排序树如下：



二叉排序树



平衡二叉树

2．现有线性表的关键字集合{33，41，20，24，30，13，01，67}共 8 个数据元素，已知散列函数为 H(k)=(3k)%11，用开放定址法解决冲突，且 d1=h(k)，di=(di-1+(7k)%10+1)%11(i=2,3,……)，

①试在 0～10 的散列地址空间中构造散列表，并计算出等概率情况下查找成功和查找失败的平均查找长度。

2、H(33)=0　H(41)=2　H(20)=5　H(24)=6

H(30)=2 冲突　d_1=2 H_1=(H(K)+d_1)%11=4

H(13)=6 冲突 d_1=6 H_1=(H(K)+d_1)%11=1

H(01)=3 H(63)=2 冲突　d_1=2 H_1=(H(K)+d_1)%11=4　冲突

d_2=(2+(7*63)%10+1)%11=4　H_2=(H(K)+d_2)%11=(2+4) %11=6　冲突

d_3=(4+(7*63)%10+1)%11=6　H_3=(H(K)+d_3)%11=(2+6) %11=8

33	13	41	01	30	20	24		63		

0　　1　　2　　3　　4　　5　　6　　7　　8　　9　　10

等概率情况下查找成功的平均查找长度=(1*5+2*2+1*4)/8=13/8

第十章　排序

一．填空题

１．排序是将一组任意排列的数据元素按__　___的值从小到大或从大到小重新排列成有序的序列。

２．在排序前，关键字值相等的不同记录间的前后相对位置保持___　的排序方法称为稳定的排序方法。

３．在排序前，关键字值相等的不同记录间的前后相对位置_____的排序方法称为不稳定的排序方法。

４．外部排序是指在排序前被排序的全部数据都存储在计算机的_____储器中。

５．下列程序是按关键字的值从大到小进行直接选择排序的算法，将算法补充完整。

```
Select(list r,int n)
{for(i=1;_____;i++)
{ k=i；
for(j=i+1;_____;j++)
if(r[k].key<r[j].key) _____;
if(_____)
swap(r[k],r[i]);          /*r[k]与 r[i]交换*/
}
}
```

６．将下列按关键字值从小到大进行冒泡排序的算法补充完整。

```
Bubblesort(int n,list r)
{ flag=_____;
m=n-1;
while(m>0 && flag)
{flag=0;
for(i=1;i<=m;i++)
if(r[i].key_____r[i+1].key)
{flag=_____;
swap(r[i],r[i+1]);     }
m--; }}
```

７．若快速排序算法中完成一趟快速排序的算法是 Quickpass(SqList &r,int low,int high)，将完整的快速排序递归算法补充完整。

```
Quicksort(SqList &r,int s,int t)
{if(s<t)
{    i=Quickpass(L,s,t);
Quicksort(r,_____    ,_____);
```

Quicksort(r,_____); }

二．选择题

１．直接插入排序的方法是从第_____个元素开始，插入前边适当位置的排序方法。

A．1　　B．2　　C．3　　D．n

２．冒泡排序的方法是_____的排序方法。

A．稳定　　B．不稳定　　C．外部　　D．选择

３．用快速排序的方法对 n 个数据进行排序，首先将第____个元素移到在排序后的位置。

A．1　　B．2　　C．n-1　　D．n