NOI’95 “同创杯”全国青少年信息学（计算机）奥林匹克竞赛

分区联赛初赛试题（高中组） 竞赛用时：2 小时

答 题 要 求

一、全部试题答案均应写在答卷纸上，写在试卷纸上一概无效。
二、算法描述中，可以使用下列过程、函数或算符：
（1） 算术运算：＋，－，×，／，DIV，MOD
整数除（DIV）：是取二整数相除的商的整数部分。如： 11 DIV 2 = 5
取模（MOD）：是取二整数相除的余数。 如：11 MOD 2 = 1
（2） 关系运算：>，<，=，<>，>=，<=
（3） 逻辑运算：AND，OR，NOT
（4） 函数：
ABS(X)：求 X 的绝对值。如：ABS（3.14）=3.14 ABS(-3.14)=3.14
SQR(X)：求 X 的平方值。如：SQR（3）=9 SQR（-15）=225
SQRT(X)：求 X 的平方根值。如：SQRT(9)=3 SQRT(225)=15
TRUNC(X)：去掉 X 的小数部分：如 TRUNC(6.3)=6 TRUNC(-7.9)=-7
ROUND(X)：函数值是小数四舍五入后的整数值。
如：ROUND(3.14)=3 ROUND(3.16)=4 ROUND(-3.14)=-4
ORD(X)：函数值是字符在 ASCII 码中的序号。
如：ORD(‘A’)=65 ORD(‘B’)=66 ORD(‘Z’)=90 ORD(‘0’)=48
CHR(X)：X 表示 ASCII 码中的序号，函数值是该序号代表的字符值。
如：CHR(48)=’0’ CHR(65)=’A’ CHR(90)=’Z’
（5） 过程：
DEC(A,[X])：变量递减，A 为有序变量，X 缺省时为 1。
INC(A,[X])：变量递增，A 为有序变量，X 缺省时为 1。

一、基础题：

<1> 执行① C>DIR 命令后，屏幕上显示如下画面：

```
FORMAT   COM   12145
SYS      COM    4878
PUC      BAT     126
XCOPY    EXE   11216
4 FILE（S） 123456  bytes  free
```

接着又顺序执行了如下几条 DOS 命令：

② C>DIR> DF.TXT //表示将列表显示的目录作为文件写盘 //

③ C>TYPE DF.TXT

④ C>DIR

试问：执行命令③和④ 在屏幕上显示的结果是否与①相同?

<2> 列举一个问题，使问题的解能对应相应的算法。

例如对算法：　　X：=10；

Y：=5；

READ（M，N）；

S：=X*M-Y*N；

可列举出如下的问题：

学生答题，答对一题可得 10 分，答错一题则要扣去 5 分，输入答对的题数（M）与答错的题数（N），求最后得分（ S）是多少？

现有以下算法：　　K：=0 ；

FOR I：=0 TO 10 DO

K：=K+（50-I*5）DIV 2+1

请列出一个相应的问题。

<3> 有标号为 A、B、C、D 和 1、2、3、4 的 8 个球，每两个球装一盒，分装 4 盒。标号为字母的球与标号为数字的球有着某种一一对应的关系（称为匹配），并已知如下条件：

1　匹配的两个球不能在一个盒子内。

2　2 号匹配的球与 1 号球在一个盒子里。

3　A 号和 2 号球在一个盒子里。

4　B 匹配的球和 C 号球在一个盒子里。

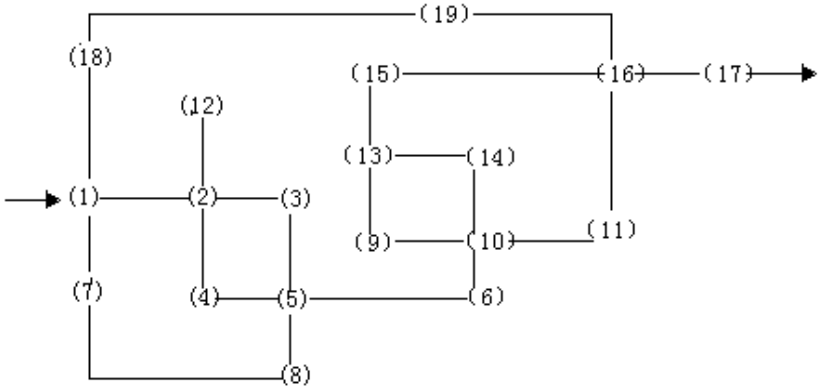
5　3 号匹配的球与 A 号匹配的球在一个盒子里。

6　4 号是 A 或 B 号球的匹配球。

7　D 号与 1 号或 2 号球匹配。

请写出这四对球匹配的情况。

<4> 从入口（1）到出口（17）的可行路线图中，数字标号表示关卡：



现将上面的路线图，按记录结构存储如下：

1　2　3　4　5　6　7　8　9　10　11　12 13　14　15　16　17　18

No	1	2	18	7	3	12	4	19	8	5	13	16	6	14	15	9	17	…
PRE	0	1	1	1	2	2	2	3	4	5	6	8	10	11	11	11	12	…

请设计一种能从存储数据中求出从入口到出口经过最少关卡路径的算法。

二、根据题目要求，补充完善以下伪代码程序：

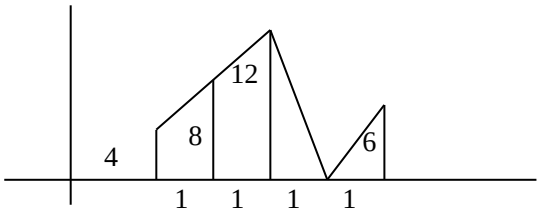
<1> 求出二个整形数组错位相加的最大面积。

1．数组面积的定义：（限定数组头尾不为 0）

设有一个数组 C=（4，8，12，0，6）

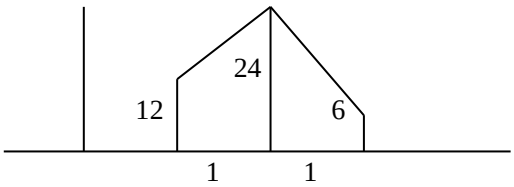
则 C 的面积为：

Sc=(4+8)/2 + (8+12)/2 + 12/2 + 6/2



也就是说，Sc=各梯形面积之和（其中梯形的高约定为 1，三角形作为梯形的特殊情况处理）。

又如 D=(12, 24, 6)是，其面积的定义为



Sd=(12+24)/2 + (24+6)/2

2．数组错位相加的定义

设有 2 个正整数的数组 a，b，长度为 n，当 n=5 时：

a=(34,26,15,44,12)　　b=(23,46,4,0,18)

对 a、b 进行错位相加，可能有下列情况

```
      34  26  15  44  12
 +)          23  46  4  0  18
-------------------------------
      34  26  15  44  12  23  46  4  0  18
```

或：

```
      34  26  15  44  12
 +)          23  46  4  0  18
-------------------------------
      34  26  15  44  35  46  4  0  18
```

或：

```
      34  26  15  44  12
 +)          23  46  4  0  18
-------------------------------
      34  26  15  67  58  4  0  18
```

或：……

最后有：

```
                  34  26  15  44  12
 +)  23  46  4  0  18                -
-------------------------------
      23  46  4  0  18  34  26  15  44  12
```

可以看到：由于错位不同，相加的结果也不同。

程序要求：找出一个错位相加的方案，使得输出的数组面积为最大。

[算法提要]： 设 a，b 的长度为 10，用 a,b: array[1..10] of integer 表示，其结果用数组 C,D: array[1..30] of integer 表示。

错位相加的过程可以从开始不重叠，然后逐步重叠，再到最后的不重叠。

梯形面积的计算公式为：(上底+下底)×高÷2
其中由于约定高为 1，故可写为(上底+下底)÷2。

程序： n = 10;

```
Function sea : real; {计算数组 C 面积}
    Begin
     J1 := 1;
     While _______①______ do
      j1 := j1 + 1;
     ENDWHILE;
     If   j1 = 3 * n   then sea := 0
              Else begin
                 J2 := 3 * n;
                 While _______②______  do
                  j2 := j2 - 1;
                  If  j1 = j2  then sea := 0
                          Else begin
                              J3 := c[j1] + c[j2];
                              For  j4 := j1 + 1 to j2 - 1 do
                                INC(j3,c[j4]*2);
                              ENDFOR;
                              Sea := j3 / 2
                             end
                    ENDIF;
                End;
```

//主程序//

```
For  i := 1 to n do read(a[I]); endfor;
For  j := 1 to n do read(b[j]); endfor;
__________③_____________;
for  i := 1 to 2 * n + 1 do
   for  j := 1 to 3 * n do _________④__________     endfor;
     for  j := 1 to n do c[j + n] := a[j]              endfor;
     for  j := 1 to n do
        _________⑤__________;
     endfor;
     p := sea;
     if p > s then begin
                d := c;
                s := p
              end;
    endif;
  endfor;
  for I := 1 to 3 * n do write(d[I],' ');   endfor;
```

write(s);
End. //主程序结束//

<2> 表的操作：设有一个表，记为 L=（a_1, a_2, …, a_n），其中：

L：表名

a_1, a_2, …, a_n为表中的元素

当 a_i为 0~9 数字时，表示元素，a_i为大写字母时, 表示是另一个表，但不能循环定义。

例如下列表的定义是合法的。（约定 L 是第一个表的表名）

L=(1,3,K,8,0,4)

K=(3,P,4,H,7)

P=(2,3)

H=(4,0,5,3)

程序要求：当全部表给出之后，求出表中所有元素的最大元素，以及表中全部元素的和。

[算法提要]：表用记录类型定义：

长度（LENGTH）

表体（是元素为字符类型的数组 ELEMENT）

队列用数组 BASE 表示；

队列指针用整型变量 FRONT 与 REAR。

为此，设计一个字符入队的过程 inqueue，出队函数 outqueue，表中最大元素及元素求和均采用递归计算。

程序：

```
PROCEDURE INQUEUE(Q，C);    //过程需要二个参数，Q 记录类型，C 字符类型//
  Q.REAR := _________①_________;
  Q.BASE[Q.REAR] := C；
   END;  //过程结束//

FUNCTION OUTQUEUE(Q)    //函数需要一个参数，Q 记录类型//
  Q.FRONT := _________②_________;
  OUTQUEUE := Q.BASE[Q.FRONT]
   END；  //函数结束//

FUNCTION MAXNUMBER(C)  //函数需要一个参数，C 字符类型//
   Max := CHR(0);
   FOR i:=1 to T[C].LENGTH  DO
       CH := T[C].ELEMENT[i];
       IF _______③________ THEN
                    M := MAXNUMBER(CH)
                 ELSE
                    M := CH
      ENDIF;
       IF   MAX < M  THEN
            MAX := M
      ENDIF;
   ENDFOR;
    __________④___________
```

```
END;  //函数结束//

FUNCTION  TOTAL(C)    //函数需要一个参数，C：字符类型//
  K := 0;
  FOR i:= 1 TO T[C].LENGTH  DO
     CH := T[C].ELELMENT[i];
      IF _________⑤__________ THEN
                          M := TOTAL(CH);
                        ELSE
                          M := ORD(CH)-ORD('0');
       ENDIF
       K := K + M
  ENDFOR;
  TOTAL := K;
   END;  //函数结束//

//主程序//
   MAX := 36;
 FOR TABNO := 'A' TO 'Z' DO
    T[TABNO].LENGTH := 0;
 ENDFOR ;
 Q.FRONT := 0;  Q.REAR := 0;
  INQUEUE(Q,'L');
  WHILE (Q.FRONT <>Q .REAR ) DO
      TABNO := OUTQUEUE(Q);
      WRITE(TABNO, '=');
      READLN(S);
      i := 1;
       WHILE S[i] <> '(' DO
         i := i+ 1;
       ENDWHILE;
       WHILE S[i] <> ')' DO
           IF (S[i]>='A') AND (S[i]<='Z')  THEN
            S[i]:=CHR(ORD(S[i])+ORD('A')-ORD('a'));
            IF (S[i]>='A') AND (S[i]<='Z') THEN
               INC(T[TABNO].LENGTH);
                  T[TABNO].ELEMENT[T[TABNO].LENGTH] := S[i];
                  INQUEUE(Q, S[i]);
            ENDIF;
        ELSE
          IF (S[i]>='0') ANDN (S[i]<='9') THEN
                   INC(T[TABNO].LENGTH);
                   T[TABNO].ELEMENT[T[TABNO].LENGTH] := S[i]
          ENDIF;
```

```
            INC(i)
          ENDIF;
       ENDWHILE;
     ENDWHILE;
  WRITE('The max number in table L is:', maxnumber('L'));
  WRITE('Total  is:', total('L'))
  END.  //主程序结束//
```

<3> 设有一个实数，以字符串形式存放于数组 x 中，用 x：array[1..N]of char 表示。其中 x[1]若为'-'，表示负数；若为'+'、'.'或' '，则表示正数。若为数字，也认为是正数。

例如　x=(' ','2','0',' ','3','.','5','%')　则表示 203.5

x=('-','1','.',' ','2','0','%')　则表示-1.2

约定：在字符串 x 中，除 x[1]外，其后可以包含有若干个'.'与' '，但仅以第一次出现的为准，空格不起任何作用，并以字符'%'作为结束标志。

程序要求：将输入的字符串还原成实数输出（小数点后无用的 0 应除去），还原的结果以下列形式存放（不需要输出）。

F：数符。正数放 0，负数放 1。

A：array[1..N] of integer; 存放数字，不放小数点。

K：表示 A 中有效数字的个数。

J：表示小数点后的位数。

例如：数 203.24，还原后结果的存放是：

F=0

A=(2, 0, 3, 2, 4)

K=5

J=2

又如：数-33.0740，还原后结果的存放是：

F=1

A=(3, 3, 0, 7, 4)

K=5

J=3

[算法提要]：x : array[1..10] of char;可放长度定为 10；首先读入字符串，然后处理数的符号，在还原的过程中，需要判定整数部分与小数部分，同时去除多余的空格和小数点，并约定输入是正确的，不用作出错检查。

程序：

```
FOR I := 1 TO 10 DO A[I] := 0;      ENDFOR;
 FOR I := 1 TO 10 DO READ(X[I]);   ENDFOR;
 J := 0; F := 0; K := 0; B := 0;
 IF X[1] = '-' THEN BEGIN
              _____①_____
              _____②_____
            END
       ELSE IF X[1] := ' ' THEN I := 2
                   ELSE I := 1;
            ENDIF;
```

```
    ENDIF;
    WHILE  _________③_________  DO
        I := I + 1;
    ENDWHILE
    WHILE  ___________④____________ DO
     IF  (X[I] >= '0')  AND  (X[I] <= '9')   THEN
                                K := K + 1;
                                   _________⑤_____________;
                                  IF   B = 1  THEN
                                                _______⑥_________
                                  ENDIF
            ELSE  IF (X[I]='.') AND (B=0) THEN
                                B := 1;
                ENDIF
            I := I + 1
      ENDIF;
   ENDWHILE;
   IF   J > 0   THEN  WHILE A[K]=0  DO
                        __________⑦_________
                        __________⑧_________
                  ENDWHILE;
   ENDIF.
END.   //程序结束//
```