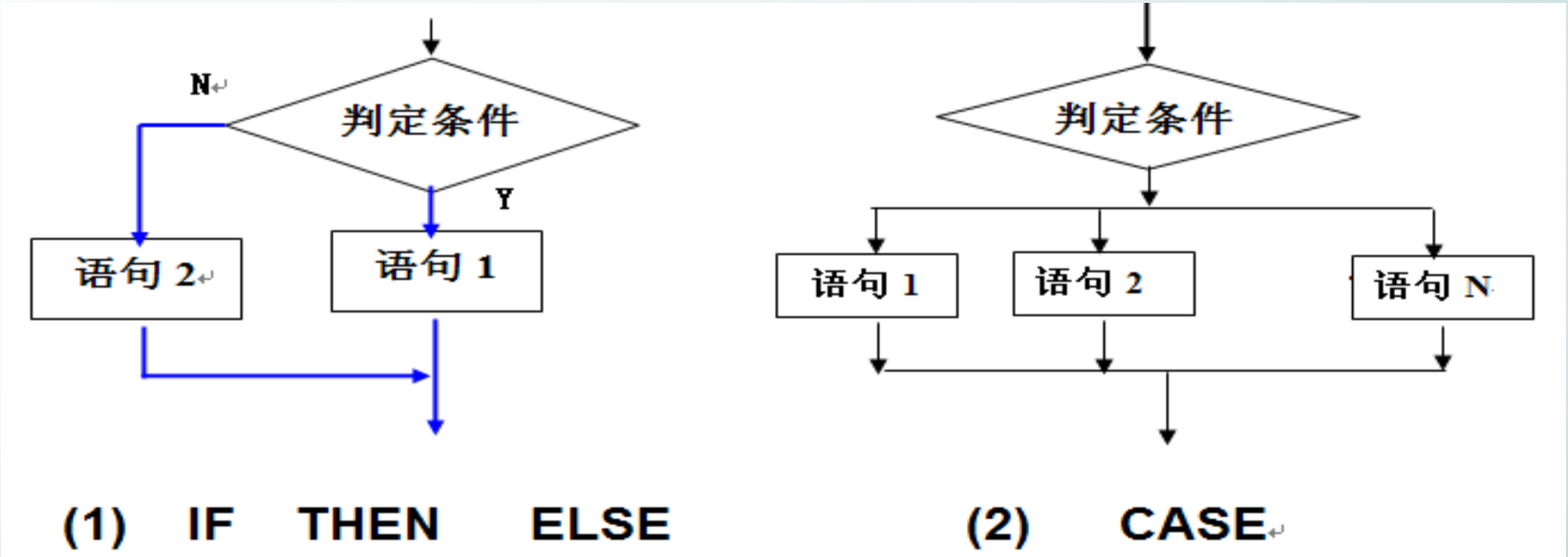
7.1 分支程序设计

- 7.1.1 分支程序结构
- 7.1.2 单分支程序
- 7.1.3 复合分支程序
- 7.1.4 多分支程序

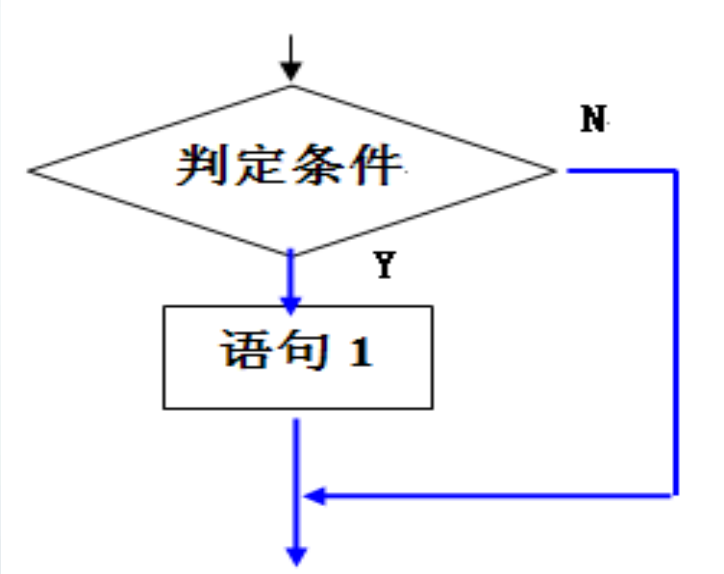
7.1.1 分支程序结构

- 分支程序结构有二种形式：两个分支和多个分支。

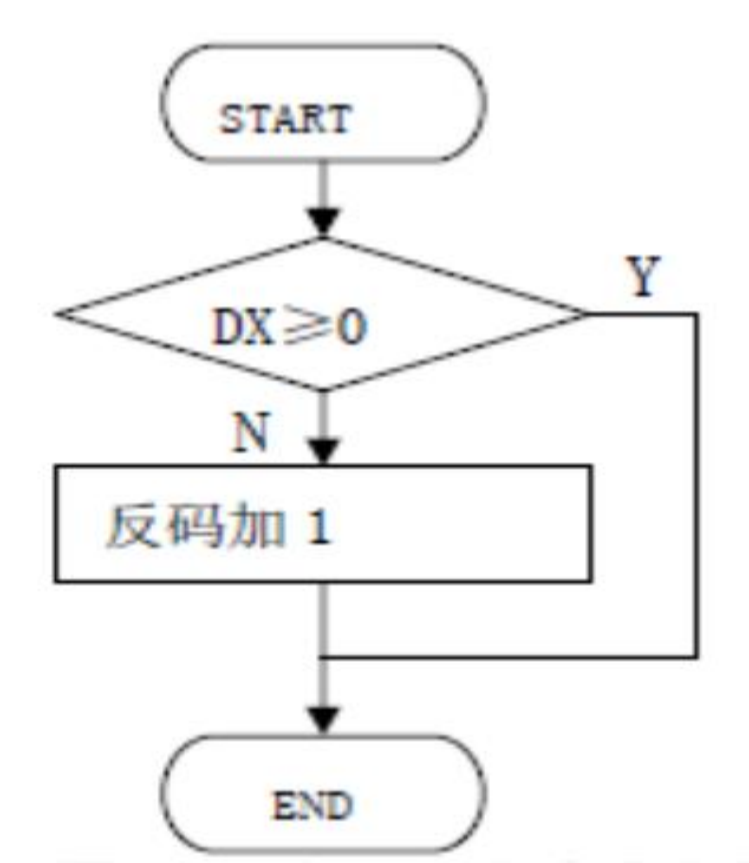


7.1.2 单分支结构程序

- 单分支结构程序（IF--THEN）：是分支结构程序的最简单形式。



- 例　双字长数存放在DX和AX寄存器中(高位在DX)，求该数的绝对值(用16位指令)
  - 算法分析：
    - 1. 双字长数高字在DX中，低字在AX中；
    - 2. 判该数的正负，为正数（最高位为0），该数不处理；为负数，就对该数求补（即反码加1）。




```
code  segment
        assume  cs:code
start:
        test   dx，8000h ；测试数的正负
        jz     exit              ；不为负数就退出
        not    ax
        not    dx
        add    ax，1
        adc    dx，0
exit:
        mov    ah, 4ch
        int    21h
code  ends
        end    start
```

7.1.3 复合分支程序

- 如果在分支结构中又出现分支，这就是复合分支结构。

❖ 例　从键盘输入一位十六进制数，并将其转换为十进制数输出显示

❖ 算法分析：

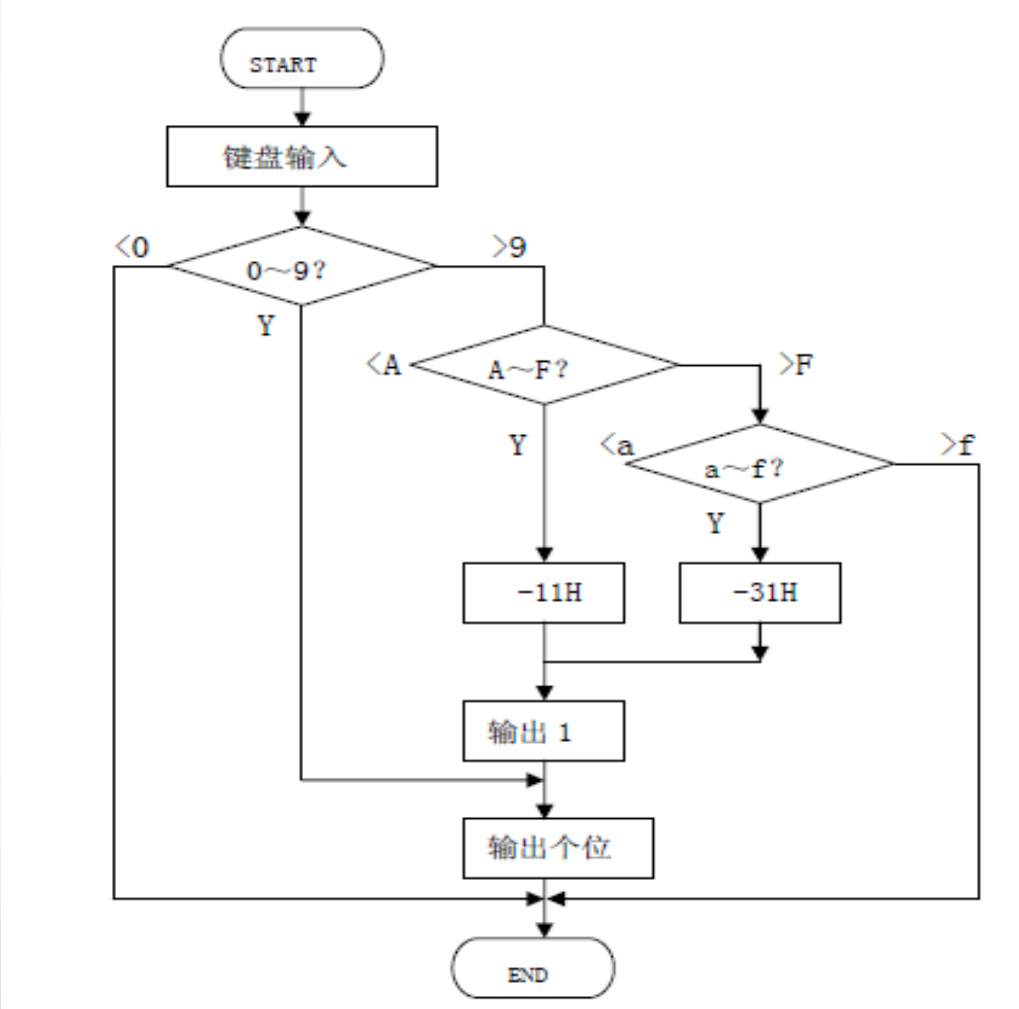
从键盘输入一个十六进制数，有以下四种情况：

1. 为数字**0～9**（**ASCII**码**30～39H**），无需处理，直接输出；

2. 为大写字母**A～F**（**ASCII**码**41～46H**），先输出**31H**，再输出该数**ASCII**码**-11H**；

3. 为小写字母**a～f**（**ASCII**码**61～66H**），先输出**31H**，再输出该数**ASCII**码**-31H**；

4. 该数不为**0～9**、**A～F**、**a～f**，是非法字符，应退出程序或输出错误信息。


START
键盘输入
<0
0～9?
>9
Y
<A
A～F?
>F
Y
<a
a～f?
>f
Y
-11H
-31H
输出 1
输出个位
END

```
code    segment
        assume  cs:code
start:  mov   ah，1              ；键盘输入
        int   21h
        cmp   al，30h
        jl    exit               ；非法输入
        cmp   al，39h
        jle   dig                ；输入是数字0～9
        cmp   al，41h
        jl    exit               ；非法输入
        cmp   al，46h
        jle   print              ；输入是大写A～F
        cmp   al，61h
        jl    exit               ；非法输入
```

```
        cmp   al，66h
        jg    exit              ；非法输入
        sub   al，31h
        jmp   out1              ；输入是小写a～f
print：sub   al，11h
out1：mov   dl，31h             ；输出字符1
        mov   ah，2
        push  ax                ；暂存AX
        int   21h               ；int指令改写了AX
        pop   ax                ；恢复AX
dig:    mov   dl，al            ；输出个位
       mov   ah，2
       int   21h
exit:   mov   ah，4ch           ；程序终止并退出
       int   21h
code    ends
        end   start
```

7.1.4 多分支程序

- 如果在分支结构中有超过两个以上的多个可供选择的分支，这就是多分支结构。

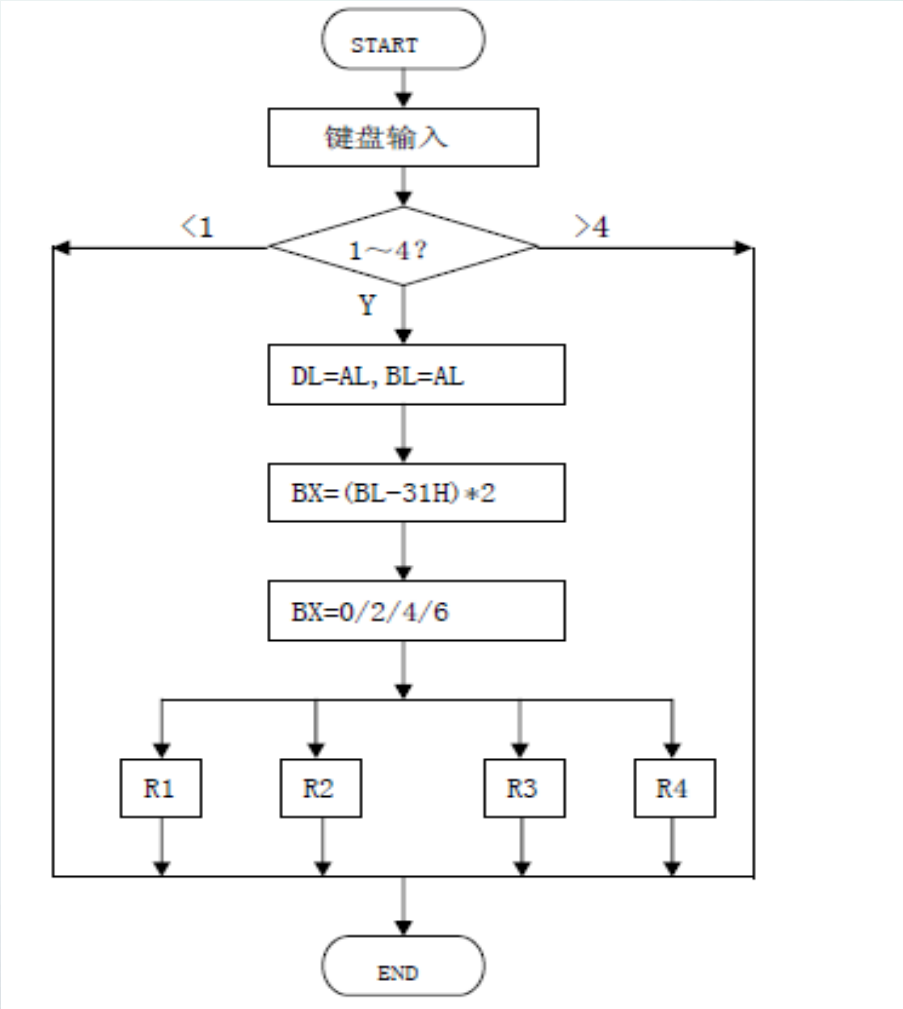
- 如果对多分支的条件逐个查询以确定是哪一个分支，只会增加代码和时间，为了尽快进入某个分支，可以采用分支向量表法。

- 例　根据键盘输入的一位数字(1～4)，使程序转移到4个不同的分支中去，以显示键盘输入的数字。

- 算法分析：从键盘输入一个数1～4，

  1. 建立一个分支向量表branch，集中存放四个分支的偏移地址；
  2. 每个偏移地址位16位，占用2个单元；
  3. 四个分支的偏移地址在转移地址表的地址是：

     转移地址表首址+输入数字（0～3）×2；
  4. 用间接寻址方式转向对应分支。


START
键盘输入
<1
1～4?
>4
Y
DL=AL, BL=AL
BX=(BL-31H)*2
BX=0/2/4/6
R1
R2
R3
R4
END

```
code    segment
        assume  cs:code，  ds:code
start:    mov   ax,code          ；  ds=cs
        mov   ds,ax
        mov   ah，  7            ；  键盘输入无回显
        int   21h
        cmp   al，  31h
        jl    exit               ；  非法输入
        cmp   al，  34h
        jg    exit               ；  非法输入
        mov   dl，  al           ；  放入dl，待显示
```

```
        mov   bl，al
        sub   bl，31h          ；转换ascii码为数值
        shl   bl，1       ；(bl)×2，指向分支向量表中某地址
        mov   bh，0
        jmp   branch[bx]       ；转向分支
r1：      mov   ah，2
        int   21h              ；显示键盘输入的数字
        jmp   exit
r2：      mov   ah，2
        int   21h
        jmp   exit
r3：      mov   ah，2
        int   21h
```

```
        jmp   exit
  r4:     mov   ah，2
        int   21h
        jmp   exit
  exit:  mov   ah，4ch；程序终止并退出
        int   21h
branch  dw    r1
        dw    r2
        dw    r3
        dw    r4
code    ends
        end   start
```

7.2 循环程序设计

- 循环指令
- 循环程序结构
- 计数循环程序
- 条件循环程序
- 条件计数循环程序
- 多重循环程序

循环指令

- **LOOP** 循环
- **LOOPZ / LOOPE** 为零或相等时循环
- **LOOPNZ / LOOPNE** 不为零或不相等时循环

- 指令：**LOOP　OPR**

  测试条件:**CX ≠ 0**，则循环
- 指令：**LOOPZ / LOOPE　OPR**

  测试条件:**ZF=1 AND CX≠0**，则循环
- 指令：**LOOPNZ / LOOPNE　OPR**

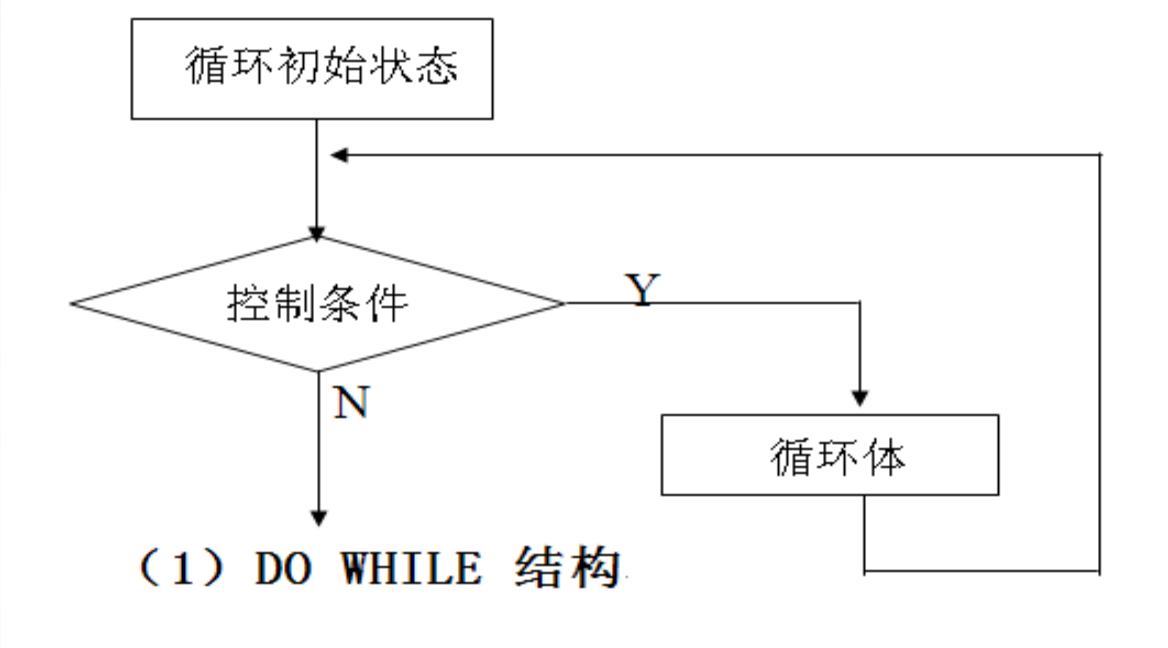
  测试条件:**ZF=0 AND CX≠0**，则循环
- 操作：首先CX寄存器减1，然后根据测试条件决定是否转移。

❖ 例　在首地址为MESS长为19字节的字符串中查找 ‘空格’(20H) 字符,如找到则继续执行，否则转标号NO。用循环指令实现程序的循环。

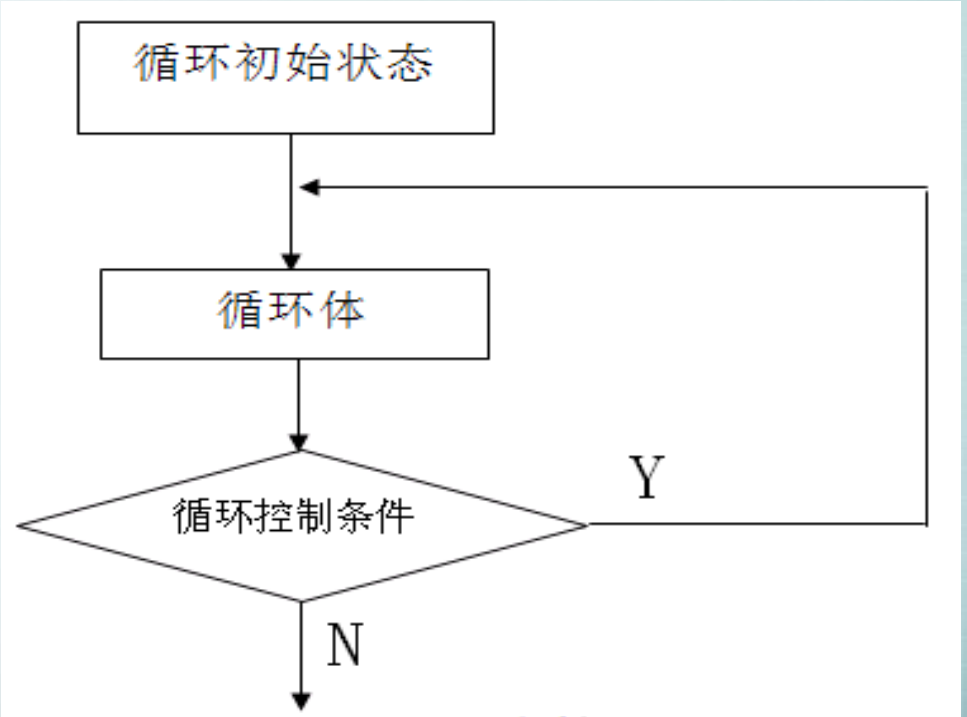
```
    MOV    AL，20H
    MOV    CX，19
    MOV    DI，-1
LK: INC    DI
    CMP    AL, MESS[DI]
    LOOPNE  LK
    JNZ    NO
    …
```

7.2.1 循环程序结构

- 循环程序有两种结构形式： **DO---WHILE结构** 和 **DO---UNTIL结构**。
- 循环程序由三部分组成：循环初始状态、循环控制、循环体。
- 循环控制条件有三类：计数循环、条件循环、条件计数循环。



（1）DO WHILE 结构



（2）DO UNTIL 结构

7.2.2 计数循环程序

- 计数循环：用循环计数器的值控制循环。

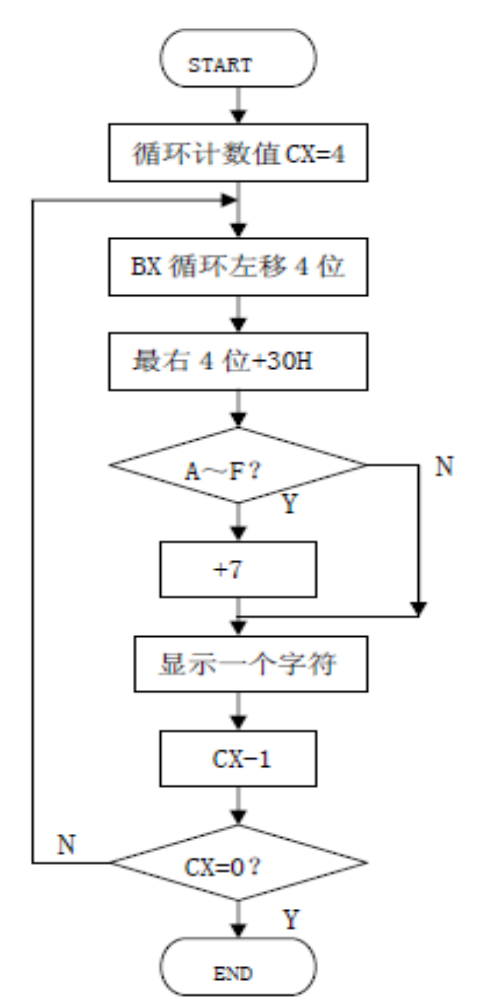
- 例7.4

❖ 例7.4　把BX中的二进制数用十六进制显示. 设BX=123AH

❖ 算法分析

1. 屏幕显示字符用2号DOS功能调用，DL=输出字符。
2. 屏幕上如何显示 BX=123AH=0001 0010 0011 1010
3. BX=1　　2　　3　　A H,

   1-->31H,　2-->32H,　3-->33H,　A-->41H
4. BX循环左移4位。




```
code    segment
    assume  cs:code
start:
    mov   cx，4
shift:
    rol   bx，1
    rol   bx，1
    rol   bx，1
    rol   bx，1
    mov   al，bl
    and   al，0fh
    add al, 30h ；转为ascii
    cmp   al，39h
```

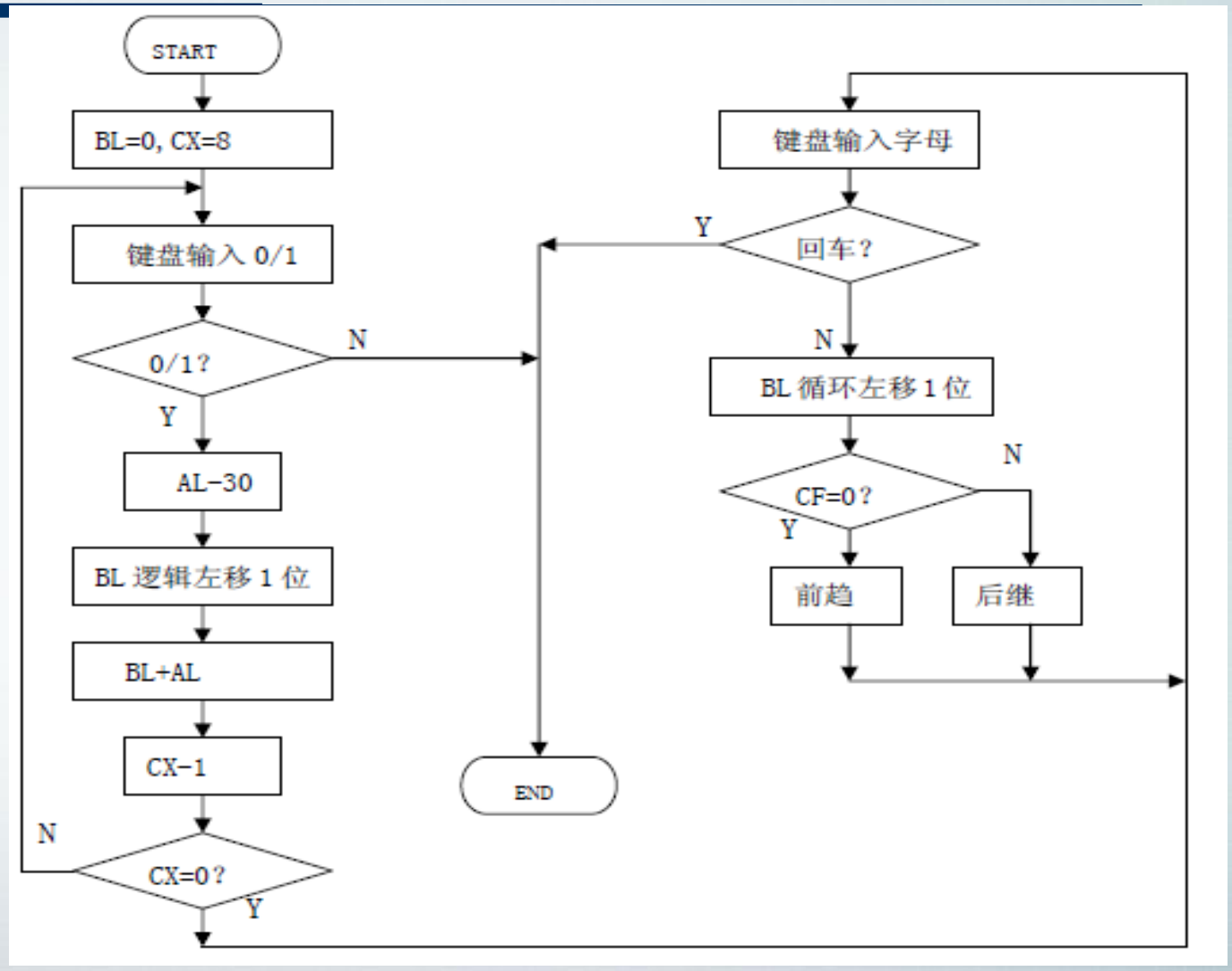
```
          jle   dig
           ；是0～9则转dig
          add   al，7
           ；是A～F
dig:   mov   dl，al
       mov   ah，2
       int   21h
       loop  shift
       mov   ah，4ch
       int   21h
code    ends
   end   start
```

7.2.3 条件循环程序

- 在循环程序中，有时候每次循环所做的操作可能不同，即循环体中有分支的情况，需要依据某一个标志来决定做何操作。标志位为1表示要做操作A，标志位为0表示要做操作B，我们可把这种标志字称为逻辑尺。

- 例：从键盘输入8位二进制数作为逻辑尺。再输入一个英文字母，根据逻辑尺当前的最高位标志显示输出该字母的相邻字符，标志位为0则显示其前趋字符，标志位为1则显示其后继字符。显示相邻字符后，逻辑尺循环左移一位，再接收下一个字母的输入，并依据逻辑尺显示相邻字符，直到回车键结束程序。

- 算法分析
  - 1. 循环次数已知，但每次循环所做的操作不同；
  - 2. 设置逻辑尺，循环中依据逻辑尺中的标志位选择操作。


START
BL=0, CX=8
键盘输入 0/1
0/1?
N
Y
AL-30
BL 逻辑左移 1 位
BL+AL
CX-1
CX=0?
N
Y
END
键盘输入字母
回车?
Y
N
BL 循环左移 1 位
CF=0?
N
Y
前趋
后继

```
code    segment
          assume  cs:code
start:    mov   bx，0                          ；初始化
          mov   cx，8
inlog:    mov   ah，1                  ；键盘输入0/1
          int   21h
          cmp   al，30h
          jb    exit                   ；非法输入
          cmp   al，31h
          ja    exit                   ；非法输入
          sub   al，30h                        ；输入是0/1
```

```
        shl   bl，1
        add   bl,al
        loop  inlog
        mov   ah，2
        mov   dl，10          ；输出换行
        int   21h
inchr：  mov   ah，1          ；键盘输入字母
        int   21h
        cmp   al，13
        je    exit             ；回车键
        mov   dl,al
```

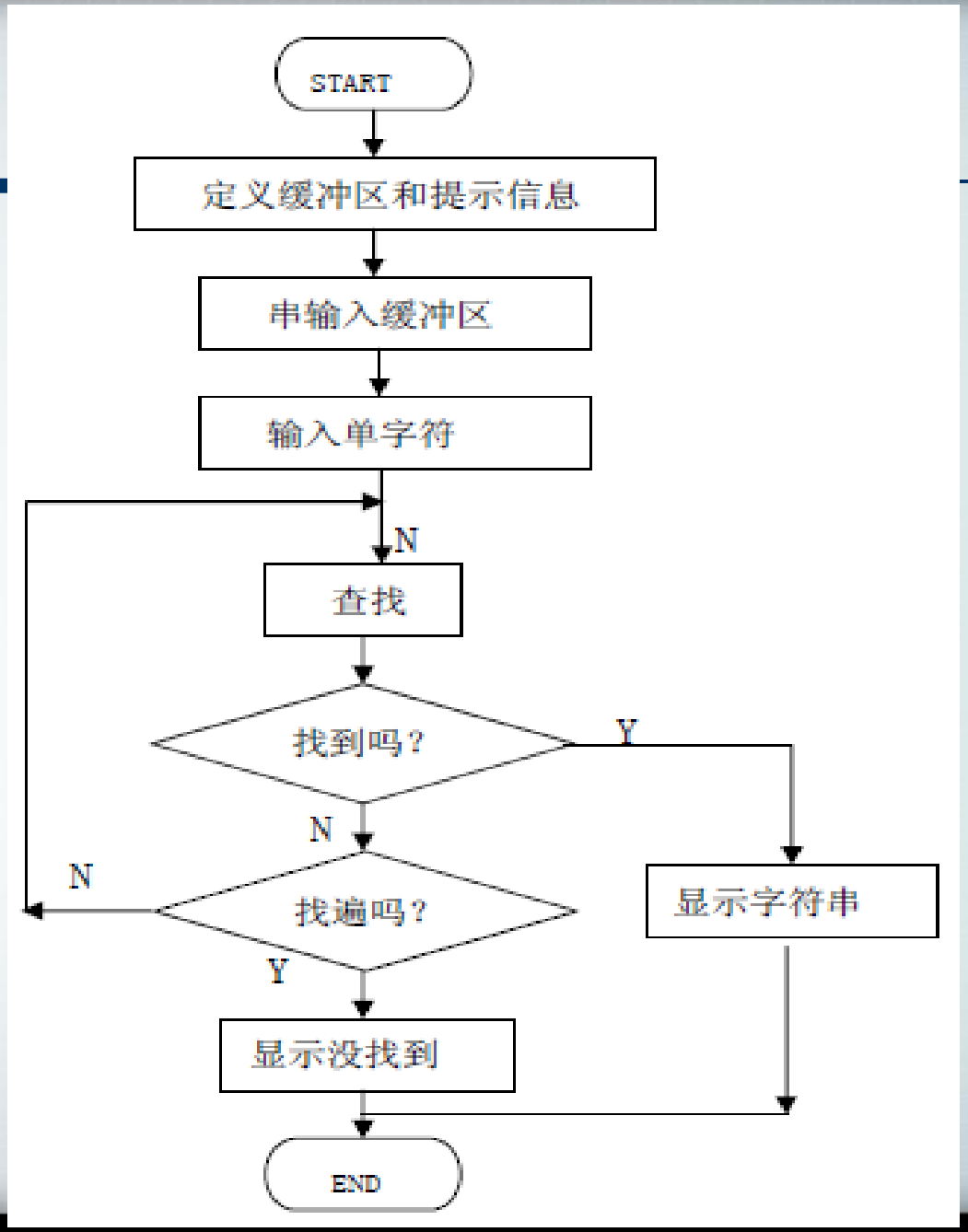
```
        rol   bl，1
        jnc   k30          ；是0 则转k30
        inc   dl
        jmp   putc
k30:    dec   dl
putc:   mov   ah，2
        int   21h
        jmp   inchr
exit:   mov   ah，4ch              ；程序终止并退出
        int   21h
code    ends
        end   start
```

7.2.4 条件计数循环程序

- 例7.6　设置键盘缓冲区为16个字节，从键盘输入一串字符，然后再从键盘输入一个单个字符，查找这个字符是否在字符串中出现，如果找到，显示该字符串，否则显示‘NOT FOUND’。

- 算法分析：

  1. 使用DOS系统功能调用10号功能实现键盘缓冲 区输入，
  2. 使用1号功能实现单个字符输入，
  3. 使用9号功能实现字符串显示输 出。
  4. 程序采用循环结构实现查找，最大计数值为实际输入的字符个数。


START
定义缓冲区和提示信息
串输入缓冲区
输入单字符
N
查找
找到吗？
Y
N
N
找遍吗？
显示字符串
Y
显示没找到
END

```
data    segment
        buffer   db   16,?,16 dup(?),13,10,'$'
        inputs   db   13，10，  'input string:$'
        getc     db   13，10，  'input char:$'
        output   db   13，10，  'not found$'
data    ends
code    segment
         assume  cs:code，ds:data
start:     mov   ax，data          ；ds赋值
           mov   ds，ax
           lea   dx，inputs        ；信息提示输入串
           mov   ah,9
           int   21h
           lea   dx，buffer        ；键盘输入串到缓冲区
```

```
        mov   ah,10
        int   21h
        lea   dx，getc              ；信息提示输入字符
        mov   ah,9
        int   21h
        mov   ah,1                  ；输入字符到al
        int   21h
        mov   bl，al                ；保存到bl
        lea   di，buffer+1          ；di作为指针指向缓冲区
        mov   cl，buffer+1         ；cx设置计数值
        mov   ch，0
seek：inc   di
        cmp   bl，[di]
        loopne   seek               ；未完且没找到，转seek继续循环
```

```
        jne   nof              ；没找到，转nof输出‘not found’
        mov   dl,10            ；输出换行
        mov   ah,2
        int   21h
        lea   dx，buffer+2              ；指向缓冲区，输出字符串
        mov   ah,9
        int   21h
        jmp   exit
nof：lea   dx，output
        mov   ah,9
        int   21h
exit：mov   ah，4ch
        int   21h
code    ends
end   start
```

7.2.5 多重循环程序

- 例
  - 显示输出20H～7EH的ASCII码字符表，每行16个字符。
  - 算法分析：
    - 1. 20H～7EH的ASCII字符共有95个，需6行显示。
    - 2. 程序需两重循环，内循环输出每行16个字符，循环计数初值为16，程序终止条件是显示最后一个字符。

高级语言程序描述

```
first=20h
last=7eh
char= first
x=1              ; 行号
y=1              ; 列号
do  while  char<last
      k=16
      do   while   k>0  and  char<last
       @ x, y  say  char
       char=char+1
              y=y+1
              k=k-1
      enddo
      x=x+1
      y=1
enddo
```

```
code    segment
    assume  cs:code
     k=16
     first=20h
     last=7eh
start:
          mov   dx,first          ；从第一个开始
a10:
          mov   cx，k             ；每行16个
a20:
          mov   ah，2             ;输出显示ASCII码
          int   21h
```

```
cmp   dl，last          ；是最后一个则退出
je    exit
push  dx                ；暂存dx
mov   dl，20h           ；空2格
int   21h
int   21h
pop   dx                ；恢复dx
add   dx，1     ；下一个要输出的ASCII码
loop  a20               ；进入内循环
push  dx                ；暂存dx
```

```
        mov   dl，13            ；回车
        int   21h
        mov   dl，10            ；换行
        int   21h
        pop   dx                ；恢复dx
        loop  a10               ；进入外循环
exit：  mov   ah，4ch
        int   21h
code    ends
end   start
```