课时6 循环指令

考点	重要程度	占分	题型
1.循环指令LOOP	★★★★★	5~10	选择 填空
2.调用	必考		大题
3.分支			

考点1　循环指令LOOP

LOOP label

;功能1：ECX←ECX－1　　相当于 → **DEC ECX**

;功能2：若ECX≠0，转移到LABEL

;否则，顺序执行　　相当于 → **JNZ label**

JECXZ

CMP ECX,0

JZ label

LOOPE/LOOPZ label

功能1：ECX←ECX－1

功能2：若ECX≠0且ZF=1，转移到LABEL

扫码观看
视频讲解更清晰

LOOPNE/LOOPNZ label

功能1：ECX←ECX一1

功能2：若ECX≠0且ZF=0，转移到LABEL

【题1】循环控制指令 LoopNZ/LoopNE 控制循环继续执行的条件是(CX≠0 且 ZF=0)

扫码观看
视频讲解更清晰

考点2　调用

1.在汇编语言中，常把子程序称为过程(procedure)

C语言中的函数是子程序，也就是汇编语言中的过程。

调用子程序过程、函数 在本质上是控制转移，它与无条件转移的区别是调用子程序要考虑返回。

处理器提供专门的过程调用指令CALL和过程返回指令RET

2.call label 段内直接调用指令

具体操作：1)把返回地址偏移EIP内容压入堆栈

2)使得EIP之内容为目标地址偏移，从而实现转移

返回地址：紧随过程调用指令的下一条指令的地址有效地址

目标地址：子程序开始处的地址有效地址

3.RET 段内返回指令

RET指令用于从子程序返回到主程序。

执行该指令时，从堆栈顶弹出返回地址，送到指令指针寄存器EIP

4.过程

（a）调用前的堆栈　（b）调用后的堆栈　（c）返回后的堆栈

执行指令时，通常利用堆栈传递入口参数，而利用寄存器传递出口参数

```
int cfg2(int x, int y)
{return （2 * x + 5 * y + 100）;}
 int main( )
{
   int val;
   val = cfg2(23, 456);
   printf("val=%d\n", val);
   return 0;
}
```

扫码观看
视频讲解更清晰

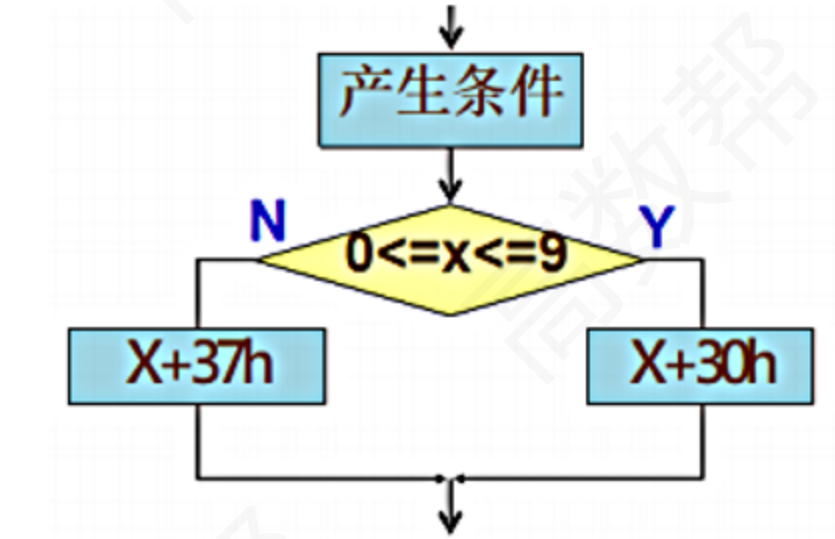
```
_main PROC ;过程开始
                ;val = cfg2(23, 456)
push 456        ;把参数y（000001c8H）压入堆栈
push 23         ;把参数x（00000017H）压入堆栈
call cfg2       ;调用函数cfg2
                ;printf("val=%d\n", val)
push eax        ;把val值压入堆栈
push OFFSET FMTS ;把输出格式字符串首地址压入堆栈
call _printf    ;调用库函数_printf
add esp, 16     ;平衡堆栈
xor eax, eax    ;由eax传递返回值
ret             ;返回
```

```
cfg2 PROC push ebp   ;把EBP压入堆栈
mov ebp, esp         ;使得EBP指向栈顶
mov eax, DWORD PTR [ebp+12]     ;从堆栈取参数
y mov ecx, DWORD PTR [ebp+8]    ;从堆栈取参数
x lea eax, DWORD PTR [eax+eax*4+100]     ;EAX=5*y+100
lea eax, DWORD PTR [eax+ecx*2]           ;EAX=EAX+2*x
pop ebp   ;恢复EBP
ret       ;返回
cfg2 ENDP
```

考点3　分支

把一位十六进制数转换为对应ASCII码，对应关系可表示为一个分段函数：

【题2】汇编语言程序设计有 顺序程序 、 分支程序 、 和 循环程序 等三种基本结构形式。



$$Y=\begin{cases} X+30H, & (0\le X\le 9) \\ X+37H,(0AH\le X\le 0FH) \end{cases}$$

```
int cff2(int m)
 {
   m = m & 0x0f; ////确保一位十六进制数（在0-15之间）
   if ( m < 10 )
        m += 0x30;
   else
        m += 0x37;
   return m;
 }
```

```
   push ebp  mov ebp, esp ;建立堆栈框架
  ;m = m & 0x0f;
   mov eax, DWORD PTR [ebp+8]
  5 and eax, 15
  6 mov DWORD PTR [ebp+8], eax
```

扫码观看
视频讲解更清晰

```
 cmp     DWORD PTR [ebp+8]
 jge SHORT LN2cff2
                          ;≤10
mov ecx, DWORD PTR [ebp+8]
add ecx, 48
mov DWORD PTR [ebp+8], ecx
jmp SHORT    LN1cff2

LN2cff2: ;≥10
        mov edx, DWORD PTR [ebp+8]
        add edx, 55
        mov DWORD PTR [ebp+8], edx

LN1cff2:          ; 结尾 return m;
        mov eax, DWORD PTR [ebp+8]
        pop ebp
        ret
```

扫码观看
视频讲解更清晰

【题3】假设数据段定义如下：

```
DA1 DW 'C', 'D'
DA2 DB 18 DUP( ? )
DA3 DW $—DA2
……
MOV BX，DA3
MOV AX，DA1
```

上述指令执行后，BX寄存器中的内容是 12H(或 18) ，AH寄存器中的内容是 00H ，AL寄存器中的内容是 43H 。

【题4】设寄存器 AL，BL，CL 中内容均为 76H ，

XOR AL，0FH

AND BL，0FH

OR CL，0FH

执行上述指令序列后，AL= 79H ， BL=06H CL=7FH 。

【题5】分析下面程序段，

MOV AL，200

SAR AL，1

MOV BL，AL

MOV CL，2

SAR AL，CL

ADD AL，BL

试问程序段执行后(BL) = 0E4H(AL) = 0DDH 。

【题6】已知数据段中定义

```
DAT1 DB 12H，34H，56H，78H
MOV CX，0
MOV AX，WORD PTR DAT1
MOV CL，DAT1+3
```

当程序段执行完后 AX=__3412H__，CX=__0078H__。

扫码观看
视频讲解更清晰