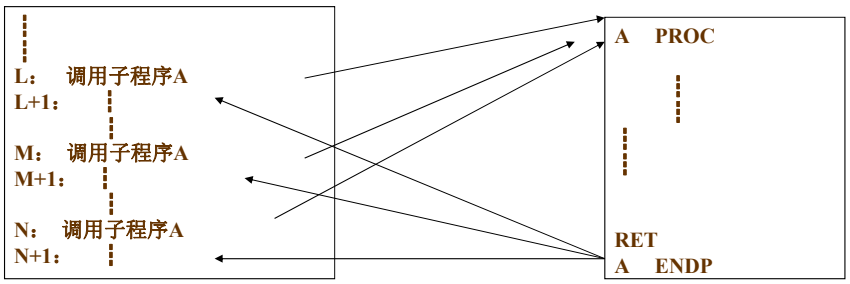
第六章 子程序设计

6．1 子程序的概念

子程序又称为过程，它相当于高级语言中的过程和函数。将重复的或经常要使用的程序段设计成可供反复调用的独立程序段，在需要时用相应的指令调用它，执行完之后再返回到调用它的程序中继续执行，这样的独立程序段称为子程序。

主程序和子程序之间的关系如图6．1所示。



6．1 子程序的概念

在设计子程序时，要考虑的主要问题如下：

（1）如何定义子程序？

（2）主程序是怎样调用子程序的？

（3）子程序执行完之后，怎样才能返回到主程序处继续往下执行？

（4）主程序每次调用子程序时提供给子程序加工的数据往往是不同的。主程序怎样把这些数据传送给子程序、而子程序又如何把加工结果交给主程序？即主程序与子程序之间用什么方式传递参数？

（5）如何编写子程序？

6．2 子程序的一般设计方法

在介绍子程序的设计方法前，先介绍子程序的调用指令、返回指令、过程的定义、现场的保护和恢复等有关内容，然后介绍子程序的设计方法。

子程序调用指令

- 1．子程序调用指令CALL
  （1）段内直接调用
  （2）段间直接调用
  （3）段内间接调用
  （4）段间间接调用
- 2．返回指令RET
  （1）语句格式：RET
  （2）语句格式：RET　n（n为偶数）

1）段内直接调用指令

格式：　CALL　　OPD

执行的操作：当操作数长度为 16位时：SP－2→ SP，IP →[SP]
OPD指定的偏移地址→ IP

当操作数长度为32位时：ESP－4→ ESP，EIP→[ESP]
OPD指定的偏移地址→ EIP

2）段内间接调用指令

格式： CALL WORD PTR OPD

执行的操作：

当操作数长度为 16位时：SP－2→SP，IP→[SP] （EA）→ IP

或 （EA）AND 0000FFFFH→ EIP

当操作数长度为32位时：ESP－4→ESP，EIP→[ESP]

（EA）→ EIP

如操作数长度的为16位时，则有效地址EA应为16位；操作数长度为32位时，EA应为32位。

3）段间直接调用指令

格式： CALL FAR PTR OPD

执行的操作：当操作数长度为 16位时：SP－2→ SP，CS→[SP]

SP－2→ SP，IP →[SP]

OPD指定的偏移地址→ IP

OPD指定的段地址→ CS

当操作数长度为32位时： ESP－4→ ESP，CS作低16位→[ESP]

ESP－4→ ESP，EIP→[ESP]

OPD指定的偏移地址→ EIP

OPD指定的段地址→CS

4）段间间接调用指令

格式： CALL DWORD PTR OPD

执行的操作：当操作数长度为 16位时：SP－2→SP，CS→[SP]

SP－2→SP，IP →[SP]

（EA）→IP

（EA+2）→CS

当操作数长度为32位时： ESP－4→ESP，CS作低16位→[ESP]

ESP－4→ESP，EIP→[ESP]

（EA）→EIP

（EA+4）的低16位→CS

小结：

- 设子程序名为**A**
- （1）段内直接调用：**CALL A**
- （2）段内间接调用：
  - **BUF DW A**
  - **LEA BX，BUF**
  - **CALL WORD PTR [BX]**
- （3）段间直接调用：**CALL FAR PTR A**
- （4）段间间接调用：
  - **BUF DD A**
  - **LEA BX，BUF**
  - **CALL DWORD PTR [BX]**

2．子程序返回指令RET

1）段内近程返回指令

格式：RET

当操作数长度为16位时：（[SP]）→IP，SP＋2→SP

当操作数长度为32位时：（[ESP]）→EIP，ESP＋4→ESP

2）段内带立即数近程返回指令

格式：RET　　n

当操作数长度为16位时：（[SP]）→IP，SP＋2→SP

SP+D16（n）→SP

当操作数长度为32位时：（[ESP]）→EIP，ESP＋4→ESP

ESP+D16（n）→ESP

2．子程序返回指令RET

（3）段间远程返回指令　　格式：RET

当操作数长度为16位时：（[SP]）→IP，SP＋2→SP

（[SP]）→CS，SP＋2→SP

当操作数长度为32位时：（[ESP]）→EIP，ESP＋4→ESP

（[ESP]）→CS，ESP＋4→ESP

（4）段间带立即数远程返回指令　格式：RET　　n

当操作数长度为16位时：（[SP]）→IP，SP＋2→SP

（[SP]）→CS，SP＋2→SP

SP+D16（n）→SP

当操作数长度为32位时：（[ESP]）→EIP，ESP＋4→ESP

（[ESP]）→CS，ESP＋4→ESP

ESP+D16（n）→ESP

6．2．2　过程的定义

过程名　　PROC　　[NEAR / FAR]

（过程体）

RET

过程名　　ENDP

（1）　如果调用的程序和子程序在同一个代码段中，则使用NEAR属性。

（2）　如果调用的程序和子程序不在同一个代码段中，则使用FAR属性。

过程体是完成某一功能的程序段。

例如：编写一子程序，将AL中的一位十六进制转换成ASCII码显示输出。

```
HEXDIS      PROC   NEAR
            AND    AL，0FH
            ADD    AL，30H
            CMP    AL，39H
            JBE    NEXT
            ADD    AL，7
NEXT：      MOV    DL，AL
            MOV    AH，2
            INT    21H
            RET
HEXDIS      ENDP
```

如果调用程序和子程序在同一代码段中，其结构形式为：

```
CODE   SEGMENT
                ⋮
               CALL   DISPLAY
                ⋮
DISPLAY        PROC   NEAR
                ⋮
               RET
DISPLAY        ENDP
CODE   ENDS
```

由于“ CALL DISPLAY ”指令和子程序DISPLAY在同一代码段中，DISPLAY应定义为NEAR属性。

6．2．2　过程的定义

如果调用程序和子程序不在同一代码段中，其结构形式为：

```
CODE1   SEGMENT
                ⋮
               CALL    FAR  PTR  DISPLAY
                ⋮
CODE1   ENDS

CODE2   PROC
                ⋮
               CALL    DISPLAY
                ⋮
DISPLAY         PROC    FAR
                ⋮
               RET
DISPLAY ENDP
                ⋮
CODE2   ENDS
```

6．2．3　现场的保护和恢复

例如：

```
SUB1  PROC
        PUSH    AX
        PUSH    BX
        PUSH    CX
        PUSH    DX
         ⋮
        POP     DX
        POP     CX
        POP     BX
        POP     AX
        RET
SUB1 ENDP
```

应该注意的是，堆栈的操作原则是后进先出。

6．2．4　子程序的参数传递方法

在编写子程序时，为了便于其他用户调用该子程序和程序的阅读，通常要写出该子程序的说明信息。说明信息一般由如下几个部分组成：

（1）子程序名
（2）功能描述
（3）入口参数
（4）出口参数
（5）使用的寄存器
（6）使用的算法和重要的性能指标
（7）其他说明

主程序与子程序之间传递参数的方法有寄存器法、约定单元法、地址表法和堆栈法。

1. 寄存器法

寄存器法就是子程序的入口参数和出口参数存放在约定的寄存器中。这种方法的优点是数据传递速度快、编程较方便、节省内存单元。其缺点是当传递的参数较多时，由于寄存器的个数是有限的，而且在处理过程中要经常使用寄存器，将导致无空闲寄存器供编写程序使用。所以，寄存器法只适用于传递参数较少的情况。

例 6.1 (DH不讲)

编写程序，将键盘输入的以非数字结束的十进制数转换成二进制数依次送BUF字缓冲区，最后以输入回车结束数据的输入。

程序清单如下：

```
STACK     SEGMENT     STACK
              DB      128  DUP (0)
STACK     ENDS
DATA    SEGMENT
BUF     DW      30  DUP (0)
DATA    ENDS
CODE    SEGMENT
                ASSUME      CS: CODE, DS: DATA, SS: STACK
START: MOV      AX, DATA
                MOV     DS, AX
                MOV     AX, STACK
                MOV     SS, AX
```

例 6.1

```
            LEA     SI, BUF  ;
L:          MOV     DI, 0
            CALL    DCB
            CMP     DI, 0
            JNZ     EXIT   ;回车则转退出
            MOV  [SI], BX
            ADD     SI, 2
            JMP     L
EXIT:       MOV  [SI], BX
            MOV     AH, 4CH
            INT     21H
```

```
DCB         PROC
            PUSH    AX
            PUSH    CX
        MOV  BX, 0
NEW :   MOV  AH, 1
        INT  21H
        CMP  AL, 0DH
        JNZ  NHC
        MOV  DI, 1
              ;回车则(DI)=1

NDIG:   POP  CX
        POP  AX
        RET
```

```
NHC:        CMP     AL, 30H
            JL      NDIG
            CMP     AL, 39H
            JG      NDIG
            AND     AL, 0FH
            CBW
            XCHG    AX, BX
            MOV  CX, 10
            MUL  CX
            ADD AX,BX
            XCHG AX,BX
            XCHG    AX, BX
            ADD     BX, AX
            JMP  NEW
DCB     ENDP
CODE    ENDS
            END     START
```

例 6.2

例6.2　　编写一主程序调用二进制数转换成十进制数显示输出的子程序，将BUF字缓冲区的数显示输出。

```
STACK SEGMENT      STACK
              DW     128  DUP (0)
   TOPS       LABEL    WORD
STACK ENDS

DATA  SEGMENT
BUF   DW     1234,-6421,807,-3400
N             =            ($-BUF)/2
STR   DB     9  DUP ( ' ' )
CHE   DW     10000, 1000, 100, 10, 1
DATA          ENDS
```

例 6.2

```
CODE          SEGMENT
               ASSUME     CS: CODE, DS: DATA, SS: STACK
START:   MOV   AX, DATA
         MOV   DS, AX
         MOV    AX, STACK
         MOV    SS, AX
         LEA   SP,TOPS
         LEA    SI, BUF
         MOV   CX, N
L:       MOV   AX, [SI]
         LEA   DI, STR
         CALL  CBD
         ADD   SI, 2
         LOOP  L
         MOV   AH, 4CH
         INT   21H
```

例 6.2　子程序说明见P172

```
CBD     PROC
        PUSH  AX
        PUSH  BX
        PUSH  CX
        PUSH  DX
        PUSH  SI
        PUSH  DI
        PUSH  DI    ; DI 是str的EA
        LEA   SI, CHE
        OR    AX, AX
        JNS   PLUS
        MOV   BYTE   PTR  [DI], ' -'
        INC   DI
        NEG   AX                  ; 将负数转为正数
```

```
PLUS:   MOV   CX, 5
  L1:   MOV   BX, [SI]
        MOV   DX, 0
        DIV   BX
        ADD   AL, 30H
        MOV   [DI], AL
        INC   DI
        ADD   SI, 2
        MOV   AX, DX
        OR    AX, AX
        JZ    L2
        LOOP  L1
  L2:   MOV   BYTE  PTR[DI], 0DH
        INC   DI
        MOV   BYTE  PTR[DI], 0AH
        INC   DI
```

```
        MOV     BYTE  PTR[DI], '$'
        POP      DX     ;     LEA       DX,STR
        MOV     AH, 9
        INT     21H
        POP     DI
        POP     SI
        POP     DX
        POP     CX
        POP     BX
        POP     AX
        RET
CBD   ENDP
CODE  ENDS
        END     START
```

2. 约定单元法

约定单元法是子程序的入口参数和出口参数存放在约定的存储单元中。这种方法的优点是子程序要处理的数据或计算的结果分别存放在各自的存储单元中。它不需要占用寄存器。其缺点是需要占用一定数量的存储单元。

例 6.3

编写程序，其主程序和被调用的子程序在同一个代码段中。要求子程序的功能是将BUF中的16位无符号二进制数转换为P进制数显示输出。其中的P为3～16中任一整数。

二进制数转换为P进制数可采用“除P取余” 法

```
   STACK      SEGMENTSTACK
              DW      128  DUP (0)
      TOPS     LABEL   WORD
   STACK      ENDS
DATA   SEGMENT
A             DW      1234H, 4352H, 3A62H, 6390H
N             =          ($-A) /2
BUF    DW     ?
STR    DB     15  DUP  (? )
P             DW      ?
J             DB      ?
DATA   ENDS
```

例 6.3

```
CODE   SEGMENT
               ASSUME    CS: CODE, DS: DATA, SS: STACK
START:         MOV     AX, DATA
               MOV     DS, AX
               MOV     AX, STACK
               MOV     SS, AX
               LEA   SP,TOPS
               MOV     CX, N
               LEA     DI, A
   L:          MOV     AX, [DI]
               MOV     BUF, AX
               MOV     P, 8
               MOV     J,  'Q'
               CALL    BCP
               MOV     P, 16
               MOV     J,  'H'
```

```
        CALL    BCP
        MOV     P, 10
        MOV     J, 'D'
        CALL    BCP
        ADD     DI, 2
        LOOP    L
        MOV     AH, 4CH
        INT     21H
BCP     PROC
        PUSH    AX
        PUSH    BX
        PUSH    CX
        PUSH    DX
        PUSH    SI
        MOV     AX, BUF
        LEA     SI, STR  ;
        MOV     BX, P
        MOV     CX, 0
```

```
L1:     MOV     DX, 0
        DIV     BX
        PUSH    DX
        INC     CX
        OR      AX, AX
        JNZ     L1
L2:     POP     AX
        CMP     AL, 10
        JB      L3
        ADD     AL, 7
L3:     ADD     AL, 30H
        MOV     [SI], AL
        INC     SI
        LOOP    L2
        MOV     AL, J
        MOV     [SI], AL
```

```
        MOV[SI+1], BYTE  PTR  0DH
        MOV[SI+2], BYTE  PTR  0AH
        MOV[SI+3], BYTE  PTR '$'
        LEA DX, STR
        MOV AH, 9
        INT 21H
        POP  SI
            POP    DX
            POP    CX
            POP    BX
            POP    AX
            RET
BCP     ENDP
CODE    ENDS
            END    START
```

- 上机练习题：
  - （参考例 **6.3** 改成有符号数，显示如下数据，每个数据按不同进制显示后空一行）
  - 数据：
  - **1234H,-5678H,3260,-4362,-765Q, 556Q**

例 6.4

例6.4　编写一程序，其主程序和子程序在同一个代码段中。要求子程序将DATA1和DATA2字节缓冲区中的N个字节的内容相加，其结果存放在SUM字节缓冲区中。

编写的程序清单如下：

```
STACK   SEGMENTSTACK
                DW      128  DUP (0)
STACK   ENDS
DATA    SEGMENT
DATA1   DB      10  DUP  (? )
DATA2   DB      10  DUP  (? )
SUM     DB      11  DUP  (0)
N               DW      10
DATA    ENDS
```

例 6.4

```
CODE    SEGMENT
                ASSUME    CS: CODE, DS: DATA, SS: STACK
START: MOV      AX, DATA
                MOV     DS, AX
                MOV     AX, STACK
                MOV     SS, AX
                CALL    MADD
                MOV     AH, 4CH
                INT     21H
```

例 6.4　子程序：MADD

```
MADD    PROC
        PUSH    AX
        PUSH    CX
        PUSH    SI
        MOV     CX, N
        XOR     SI, SI  ;  CF=0
M1:     MOV     AL, DATA1 [SI]
        ADC     AL, DATA2 [SI]
        MOV     SUM [SI] , AL
        INC     SI
        LOOP    M1
        MOV     AL, 0
        ADC     AL, 0
        MOV     SUM[ SI ], AL
```

- POP SI
- POP CX
- POP AX
- RET
- MADD ENDP
- CODE ENDS
- END START

例6.4　编写一程序，其主程序和子程序在同一个代码段中。要求子程序将DATA1和DATA2字节缓冲区中的N个字节的内容相加，其结果存放在SUM1字节缓冲区中;将DATA3和DATA4字节缓冲区中的M个字节的内容相加，其结果存放在SUM2字节缓冲区中。

编写的程序清单如下：

```
STACK   SEGMENTSTACK
                DW      128   DUP（0）
STACK   ENDS
DATA    SEGMENT
DATA1   DB      10   DUP  （? ）
DATA2   DB      10   DUP  （? ）
SUM1    DB      11   DUP  （0）
N               DW      10
DATA3   DB      100  DUP  （? ）
DATA4   DB      100  DUP  （? ）
SUM     DB      101  DUP  （0）
M               DW      100
```

```
CODE    SEGMENT
                ASSUME     CS：CODE，DS：DATA，SS：STACK
START：MOV      AX，DATA
                MOV      DS，AX
                MOV      AX，STACK
                MOV      SS，AX
                LEA      SI,DATA1
                LEA      DI,DATA2
                LEA      BX，SUM1
                MOV      CX,N
                CALL     MADD
                LEA      SI,DATA3
                LEA      DI,DATA4
                LEA      BX，SUM1
                MOV      CX,M
                CALL     MADD
                MOV      AH，4CH
                INT      21H
```

```
MADD    PROC
        PUSH    AX
        CLC
M1:     MOV     AL, [SI]
        ADC     AL, [DI]
        MOV     [BX] , AL
        INC     SI
        INC DI
        INC BX
        LOOP    M1
        MOV     AL, 0
        ADC     AL, 0
        MOV     [ BX ], AL
        POP AX
        RET
MADD ENDP
CODE ENDS
        END START
```

3．地址表法

这种方法是把参数组成一张参数表存放在某个存储区中，然后只要主程序与子程序约定好这个存储区的首地址和存放的内容，在主程序中将参数传送给地址表，在子程序中根据地址表给定的参数就可以完成其操作。

例6.5

编写一程序，其主程序和子程序在同一个代码段中。要求子程序将字节缓冲区中的多个字节的二进制内容相加，其结果存放在结果单元中。

程序清单如下：

```
DATA SEGMENT
TAB   DW   3  DUP  (0)    ; 参数地址表
BUF1  DB   10  DUP  (? )
N1          DW   6
SUM1 DW   0
BUF2  DB   20  DUP  (? )
N2          DW   12
SUM2 DW   0
DATA ENDS
```

例6.5

```
CODE      SEGMENT
      ASSUME  CS: CODE, DS: DATA, SS: STACK
START:    MOV  AX, DATA
          MOV  DS, AX
          MOV  AX, STACK
          MOV  SS, AX
      LEA   AX, BUF1          ; 将入口参数送TAB
      MOV  TAB, AX
      LEA   AX, N1
      MOV  TAB+2, AX
      LEA   AX, SUM1
      MOV  TAB+4, AX
```

```
CALL MBA                ；子程序调用
LEA  AX，BUF2           ；将入口参数送TAB
MOV  TAB，AX
LEA  AX，N2
MOV  TAB+2，AX
LEA  AX，SUM2
MOV  TAB+4，AX
CALL MBA                ；子程序调用
MOV  AH，4CH
INT  21H
```

例6.5 子程序名：MBA

```
MBA     PROC
        PUSH AX
        PUSH CX
        PUSH SI
        MOV  SI，TAB+2
        MOV  CX，[SI]
        MOV  SI，TAB
        XOR  AX，AX
M1：    ADC  AL，［SI］
        ADC  AH，0
        INC  SI
        LOOP      M1
```

例6.5 子程序名：MBA

```
        MOV  SI，TAB+4
        MOV  [SI]，AX
        POP   SI
        POP   CX
        POP   AX
        RET
MBA     ENDP
CODE    ENDS
        END      START
```

4．堆栈法

用堆栈传递参数的方法是在调用子程序之前，用PUSH指令将输入参数压入堆栈，在子程序中通过出栈方式依次获得这些参数。经过子程序操作处理后再将输出参数压入堆栈，返回主程序后再通过出栈获得结果。使用这种方式传递参数时，特别要注意堆栈中断点的保存与恢复。

例6.6

利用堆栈传递参数的方法编制一子程序，实现主程序调用不同代码段的子程序对指定的数组求和。

```
STACK     SEGMENT  PARA  STACK  'STACK'
          DB       128  DUP（？）
STACK     ENDS
DATA SEGMENT
ARYA      DB     al，a2，a3，…，an
NA    =   $－ARYA
SUMA      DW     ?
ARYB      DB     bl，b2，b3，…，bn
NB        =      $－ARYB
SUMA      DW     ?
DATA      ENDS
```

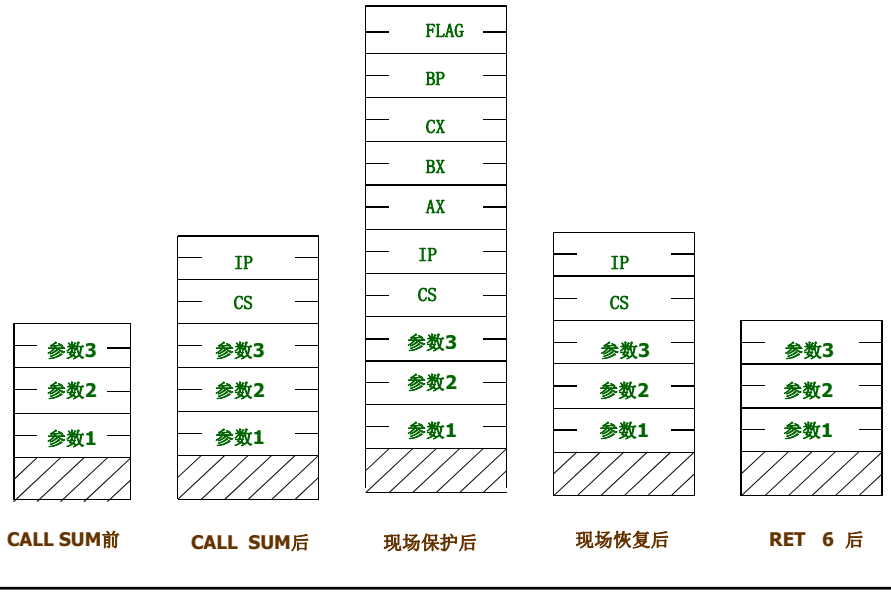
例6.6

```
CODE      SEGMENT
      ASSUME  CS：CODE，DS：DATA，SS：STACK
START：   MOV  AX，DATA
          MOV  DS，AX
          MOV   AX，STACK
          MOV  SS，AX
          MOV  AX，NA
          PUSH AX
          LEA   AX，ARYA
          PUSH AX
          LEA   AX，SUMA
          PUSH AX
```

例6.6

```
        CALL   FAR PTR SUM
        MOV  AX，NB
        PUSH AX
        LEA   AX，ARYB
        PUSH AX
        LEA   AX，SUMB
        PUSH AX
        CALL   FAR  PTR SUM
        MOV    AH，  4CH
        INT    21H
CODE    ENDS
```

例6.6程序执行堆栈变化示意图



例6.6 子程序SUM

```
PROCE     SEGMENT
ASSUME  CS：PROCE，DS：DATA，SS：STACK
SUM   PROC  FAR
          PUSH AX
          PUSH BX
          PUSH CX
          PUSH BP
          PUSHF
          MOV  BP，SP
          MOV  CX，[BP+18]
          MOV   BX，[BP+16]
          MOV  AX，0
LOPI：    ADD   AL，[BX]
```

例6.6 子程序SUM

```
          ADC    AH，0
          INC    BX      ;bx中是数的偏移量
          LOOP   LOPI
           MOV   BX，[BP+14]
          MOV    [BX]，AX
          POPF
          POP    BP
          POP    CX
          POP    BX
          POP    AX
          RET    6
SUM       ENDP
PROCE     ENDS
      END   START
```

例6.7

例6.7　设ARY1和ARY2是两个长度都为10的双字数组。请用比例变量寻址方式编写一子程序，对两个数组中的对应元素分别相乘，其结果存放在ARY3的数组中。

```
.586
STACK       SEGMENT   USE16  SATCK
        DW    128  DUP（0）
STACK       ENDS
DATA SEGMENT   USE16
        ARY1   DD   10  DUP （？ ）
        ARY2   DD   10   DUP（？ ）
        ARY3   DQ   10  DUP （？ ）  ；8字节
        TAB   DD    3   DUP （0）
DATA ENDS
```

例6.7

```
CODE  SEGMENT     USE16
        ASSUME  CS：CODE，DS：DATA, SS：STACK
START：         MOV    AX，DATA
                MOV    DS，AX
                MOV    AX，STACK
                MOV    SS，AX
                LEA    EAX，ARY1
                MOV    TAB，EAX
                LEA    EAX，ARY2
                MOV    TAB+4，EAX
                LEA    EAX，ARY3
                MOV    TAB+8，EAX   ；存储单元传递参数
                CALL   AMUL
                 MOV   AX，4C00H
                INT    21H
```

例6.7

```
AMUL  PROC
              MOV   ESI，TAB
              MOV   EDI，TAB+4
              MOV   EBP，TAB+8
              MOV   EBX，0
              MOV   CX，10
  L：         MOV   EAX，[ESI][EBX*4]
              MUL   [EDI][EBX*4]
              MOV   [EBP][EBX*8]，EAX
              MOV   [EBP+4][EBX*8]，EDX
              INC   EBX
              LOOP  L
              RET
AMUL  ENDP
              END   START
```

6．3嵌套与递归子程序

6．3．1 子程序的嵌套

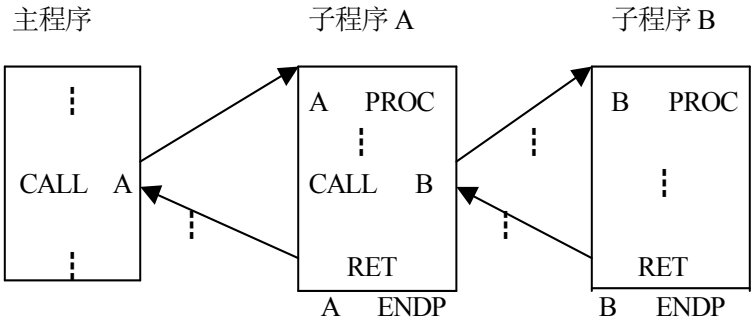


图 6．3　　子程序嵌套调用示意图

例6.8

编写一程序，要求从键盘输入0～FFFFH的十六进制正整数转换为十进制数，并将转换的结果显示输出。

这一程序由一个主程序、一个从键盘输入的十六进制正整数转换为二进制数的子程序HDC、一个二进制正整数转换为十进制数的子程序BDC、一个十进制数转换为ASCII码显示输出的子程序DISP和显示输出回车换行的子程序CRLF等部分组成。主程序可调用子程序BDC、CRLF，在BDC子程序中可再次调用DISP子程序，实现子程序的嵌套调用。编写的程序清单如下：

```
STACK SEGMENT       STACK
                DW      128  DUP（0）
STACK ENDS
DATA  SEGMENT
STR   DB   ‘是否继续输入数？需继续输入请按Y键，否则按其他键！ $’
DATA  ENDS
```

例6.8

```
CODE SEGMENT
        ASSUME   CS：CODE，DS：DATA，SS：STACK
START：     MOV  AX，DATA
            MOV  DS，AX
            MOV  AX，STACK
            MOV  SS，AX
L：         CALL  HDC   ；十六进制正整数转换为二进制数的子程序
            CALL  CRLF ；显示输出回车换行的子程序
            CALL  BDC   ；二进制正整数转换为十进制数的子程序
            CALL  CRLF
```

例6. 8

```
LEA   DX，STR
MOV   AH，9
INT    21H
MOV   AH，1
INT    21H
CMP   AL，  ‘Y’
JZ      L
CMP   AL，  ‘y’
JZ      L
  MOVAH，4CH
  INT   21H
```

子程序HDC

```
；功能描述：从键盘输入的十六进制正整数转换为二进制数送BX。
；入口参数：无
；出口参数：BX寄存器存放键盘输入的十六进制转换后得到的二进制数。
HDC   PROC
              PUSH AX
              PUSH CX
              MOV  BX，0
NKEY：        MOV  AH，1
             INT    21H
             CMP   AL，30H
             JB      EXIT ；    非十六进制数则退出
```

子程序HDC

```
CMP   AL，39H
JBE   DIGIT
CMP   AL，41H
JL    EXIT
CMP   AL，47H
JL    CHAR
CMP   AL，61H
JL    EXIT
CMP   AL，67H
JGE   EXIT
SUB   AL，20H
```

子程序HDC

```
CHAR：    SUB   AL，7
DIGIT：   SUB   AL，30H
          MOV   CL，4
          SHL   BX，CL
          MOV   AH，0
          ADD   BX，AX
          JMP   NKEY
EXIT：    POP   CX
          POP   AX
EXIT：    RET
HDC       ENDP
```

```
；功能描述：将BX寄存器中的二进制正整数转换为十进制数显示输出。
；入口参数：BX寄存器中存放二进制数。
；出口参数：十进制数显示输出。
BDC    PROC
       PUSH   CX
       MOV    CX，10000
       CALL   DISP
       MOV    CX，1000
       CALL   DISP
       MOV    CX，100
       CALL   DISP
       MOV    CX，10
       CALL   DISP
       MOV    CX，1
       CALL   DISP
       POP    CX
       RET
BDC    ENDP
```

子程序DISP

```
；功能描述：将BX寄存器中的二进制除以CX寄存器中的值，并显示输出商。
；人口参数：BX寄存器中存放二进制数被除数； CX寄存器中存放二进制数除数。
；出口参数：BX寄存器中存放BX / CX后的二进制余数。
DISP    PROC
PUSH    AX
PUSH    CX
PUSH    DX
MOV     AX，BX；
 MOV    DX，0
DIV     CX
MOV     BX，DX；将余数送BX
MOV     DL，AL；将商转换为ASCII码显示输出
ADD     DL，30H
MOV     AH，2
INT     21H
POP     DX
POP     CX
POP     AX          RET         DISP        ENDP
```

子程序CRLF

```
；子程序名：CRLF
；功能描述：显示输出回车换行。
；入口参数：无
；出口参数：无
CRLF    PROC
P USH   AX
PUSH    DX
MOV     DL，0DH
MOV     AH，2
INT     21H
MOV     DL，0AH
MOV     AH，2
INT     21H
POP     DX
POP     AX
RET
CRLF    ENDP
CODE    ENDS
                END     START
```

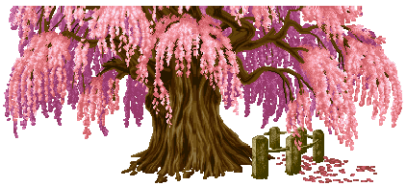
6．3．2　递归子程序（不看）

在子程序嵌套的情况下，如果一个子程序直接调用它自身，这种调用称为直接递归调用；如果一个子程序间接调用它自身，这种调用称为间接递归调用。具有递归调用功能的子程序称为递归子程序。递归调用是嵌套调用的一种特殊情况。

本章结束

练习题：

6.1、6.2、6.3、6.4、6.14、6.15、6.19



上机题

从键盘上输入学生总成绩（**10-100**个）依次存放在**TAB**缓冲区中**(**得分范围**0**～**600)**。采用子程序的结构形式编程，统计成绩为**0**≤分数≤**99**、**100**≤分数≤**199**、**200**≤分数≤**299**、**300**≤分数≤**399**、**400**≤分数≤**499** 、**500**≤分数≤**599** 、分数**=600**的人数，将统计结果以十进制形式显示输出，并将最高分送显示器输出。要求：

(1) 编写一键盘输入子程序输入学生的分数（每个分数以回车为结束）。

(2) 编写统计分数段人数的子程序（不用分数段比较方式）。

(3) 编写屏幕显示输出子程序（要求用嵌套形式，参考例**6.8**）

（**4**）编写显示回车换行的子程序；

（**5**）**2**号、**9**号、**10**号调用要求用宏定义。