3、

```
data segment
        A dw 0-21
        B dw 5
        C dw 5
        D dw 5
        E dw 2 dup(?)
data ends

code segment
        assume cs: code, ds: data
main proc far
start:
        push ds
        sub ax, ax
        push ax

        mov ax, data
        mov ds, ax

        mov ax, B
        imul C
        add ax, 5
        adc dx, 0
        mov bx, ax
        mov cx, dx
        mov ax, A
        cwd
        sub ax, bx
        sbb dx, cx
        idiv D
        mov E, ax
        mov E+2, dx

        ret
main endp
code ends
        end start
```

4、

```
data segment
        inputbuf label byte
                maxlen db 3
                actlen db ?
                scoreinput db 3 dup(?)
```

```
        numbuf db ?
        scoretab db 100 dup(?)
        tabnum db 0
        strinput db 'Enter scores: ', 13, 10, '$'
        strsort db 'After sort: ', 13, 10, '$'
        strnum db 'Number of scores: $'
data ends

code segment
        assume cs: code, ds: data
main proc far
start:
        push ds
        sub ax, ax
        push ax

        mov ax, data
        mov ds, ax

        ; get score
        lea dx, strinput
        mov ah, 09h
        int 21h
        call getscore

        ; sort score
        call sortscore

        ; show score
        lea dx, strsort
        mov ah, 09h
        int 21h
        call showscore

        ret
main endp

getscore proc near
        mov si, 0
        mov cx, 100                     ; max #input = 100
input:
        lea dx, inputbuf
        mov ah, 0ah
        int 21h                         ; get string
```

```
        cmp actlen, 0
        jz gexit
        call str2bin            ; convert string to binary, store in numbuf
        call crlf

        mov al, numbuf
        mov scoretab[si], al
        inc si
        inc tabnum
        loop input

gexit:
        ret
getscore endp

str2bin proc near
        push ax
        push bx

        mov al, scoreinput
        cbw
        sub ax, '0'
        mov bx, 10
        mul bx
        add al, scoreinput+1
        sub al, '0'
        mov numbuf, al

        pop bx
        pop ax
        ret
str2bin endp

; for (i = 0; i < n - 1; ++i)
; {
        ; for (j = i + 1; j < n; ++j)
        ; {
                ; if (a[i] < a[j])
                ; {
                        ; t = a[i];
                        ; a[i] = a[j];
                        ; a[j] = t;
                ; }
```

```
                ; }
; }

sortscore proc near
                push si
                push di
                push cx
                push ax
                push bx

                mov si, 0
                mov ch, tabnum
                sub ch, 1
sort1:
                mov di, si
                add di, 1
                mov cl, tabnum
sort2:
                mov al, scoretab[si]
                cmp al, scoretab[di]
                jnl next
                mov bl, scoretab[di]
                mov scoretab[si], bl ; swap
                mov scoretab[di], al
next:
                inc di
                mov bx, di
                cmp bl, cl
                jl sort2
                inc si
                mov bx, si
                cmp bl, ch
                jl sort1

                pop bx
                pop ax
                pop cx
                pop di
                pop si
                ret
sortscore endp

showscore proc near
                push cx
```

```
        push si
        push bx

        mov cl, tabnum
        mov ch, 0
        mov si, 0
showtab:
        mov bl, scoretab[si]
        mov bh, 0
        call bin2dec
        inc si
        loop showtab

        lea dx, strnum
        mov ah, 09h
        int 21h
        mov bl, tabnum
        mov bh, 0
        call bin2hex

        pop bx
        pop si
        pop cx
        ret
showscore endp

bin2dec proc near
        push cx

        mov cx, 10d
        call decdiv
        mov cx, 1d
        call decdiv
        call crlf

        pop cx
        ret
bin2dec endp

decdiv proc near
        push ax
        push dx

        mov ax, bx
```

```
                mov dx, 0
                div cx
                mov bx, dx
                mov dl, al
                add dl, '0'
                mov ah, 02h
                int 21h

                pop dx
                pop ax
                ret
decdiv endp

crlf proc near
                push ax
                push dx

                mov dl, 13
                mov ah, 02h
                int 21h
                mov dl, 10
                mov ah, 02h
                int 21h

                pop dx
                pop ax
                ret
crlf endp

bin2hex proc near
                push cx
                push dx
                push ax

                mov ch, 4
rotate:
                mov cl, 4
                rol bx, cl
                mov dl, bl
                and dl, 0fh
                add dl, '0'
                cmp dl, '9'
                jng printit
                add dl, 'A'-'9'-1
```

```
printit:
        mov ah, 02h
        int 21h
        dec ch
        jnz rotate
        call crlf

        pop ax
        pop dx
        pop cx
        ret
bin2hex endp

code ends
        end start
```

5、

```
data segment
        count db 1
        num db 0
        str0 db '1', 13, 10, '$'
        str1 db '2', 13, 10, '$'
        str2 db '3', 13, 10, '$'
        str3 db 'Go!', 13, 10, '$'
data ends

code segment
        assume cs: code, ds: data
main proc far
start:
        push ds
        sub ax, ax
        push ax

        mov ax, data
        mov ds, ax

        mov al, 1ch
        mov ah, 35h
        int 21h
        push es
        push bx

        push ds
        mov ax, seg counter
```

```
        mov ds, ax
        mov dx, offset counter
        mov al, 1ch
        mov ah, 25h
        int 21h
        pop ds

        in al, 21h
        and al, 11111110b
        out 21h, al
        sti

        mov si, 10000
delay:
        mov di, 10000
delay1:
        dec di
        jnz delay1
        dec si
        jnz delay

        cli
        pop dx
        pop ds
        mov al, 1ch
        mov ah, 25h
        int 21h

        ret
main endp

counter proc near
        push ds
        push ax
        push dx
        sti

        mov ax, data
        mov ds, ax

        dec count
        jnz exit
        mov count, 18
        cmp num, 4
```

```
        jnz next1
        mov num, 0
        jmp next4
next1:
        cmp num, 3
        jnz next2
        lea dx, str3
        jmp next
next2:
        cmp num, 2
        jnz next3
        lea dx, str2
        jmp next
next3:
        cmp num, 1
        jnz next4
        lea dx, str1
        jmp next
next4:
        lea dx, str0
next:
        mov ah, 09h
        int 21h
        inc num

exit:
        cli
        pop dx
        pop ax
        pop ds
        iret
counter endp
code ends
        end start
```