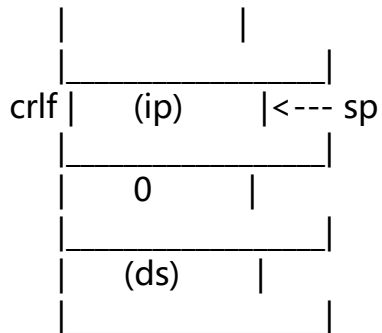
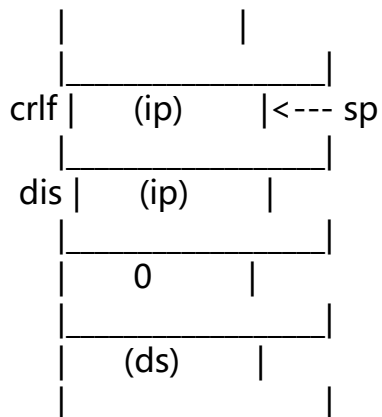


1、

(1)



(2)



(3)

2016 Happy New Year!

Total:20

Number:04

Upper:03

Lower:09

Other:04

(4)

14 01 00 00 00 64 14 32

2、

datasg segment

three db 3

mess db 'month?', 13, 10, '\$'

monin label byte

max db 3

; 1)

act db ?

mon db 3 dup(?)

alfmon db '???', 13, 10, '\$'

montab db 'JAN', 'FEB', 'MAR', '444', '555', '666', '777', '888', '999', '10 ',

'11 ', '12 '

datasg ends

codesg segment

main proc far

assume cs: codesg, ds: datasg, es: datasg

start:

push ds

sub ax, ax

push ax

mov ax, datasg

mov ds, ax

mov es, ax ; 2)

lea dx, mess

mov ah, 09h

int 21h

lea dx, monin

mov ah, 0ah

int 21h

mov dl, 13

mov ah, 02

int 21h

mov dl, 10

mov ah, 02

int 21h

cmp act, 0

je exit

mov ah, 30h

cmp act, 2 ; 3)

je two

mov al, mon

jmp conv ; 4)

two:

mov al, mon+1

mov ah, mon

conv:

xor ax, 3030h

cmp ah, 0 ; 5)

jz loc

sub ah, ah

add al, 10 ; 6)

loc:

lea si, montab

sub ax, 1 ; 7)

mul three

add si, ax

```

        mov cx, 3                ; 8)
        cld                      ; 9)
        lea di, alfmon
        rep movsb                ; 10)
        lea dx, alfmon
        mov ah, 09h
        int 21h
        jmp start
exit:
        ret
main endp
codesg ends
        end start

```

3、

```

data segment
    A dw 5
    B dw 10
    C dw 10
    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, A
    imul B
    add ax, C
    adc dx, 0
    idiv D
    add ax, 15
    mov E+2, 0
    adc E+2, 0
    mov E, ax

```

```
        ret
main endp
code ends
        end start
```

```
4、
data segment
        array dw 3, 1, 3
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, array
        cmp ax, array+2
        jne cmpAC
        cmp ax, array+4
        je show2
        jmp show1
```

```
cmpAC:
        cmp ax, array+4
        je show1
        mov ax, array+2
        cmp ax, array+4
        je show1
        mov dl, '0'
        jmp show
```

```
show1:
        mov dl, '1'
        jmp show
```

```
show2:
        mov dl, '2'
```

```
show:
        mov ah, 02h
        int 21h
        mov dl, 13
```

```

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

        ret
main endp
code ends
        end start

```

5、

```

data segment
        count db 1
        timer dw 0ffffh
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

; Here goes main procedure

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 bx

mov ax, data

mov ds, ax

sti

dec count

jnz exit

mov count, 18

inc timer

mov bx, timer

call bin2dec

exit:

cli

pop bx

pop ax

```
        pop ds
        iret
counter endp
```

```
bin2dec proc near
    push cx

    mov cx, 10000d
    call decdiv
    mov cx, 1000d
    call decdiv
    mov cx, 100d
    call decdiv
    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
```

```
code ends
end start
```