

Gestione delle Subroutine con ASM 80x86 su EMU8086

```

name NomeProg ; output file name (max 8 chars).
org 0100h
main: :::::::::::
      assegna i valori ai registri parametri di input della sub
      call NomeSub
      utilizza i valori restituiti dalla sub tramite uno o più registri
      :::::::::::
      ret ; return to the operating system
;-----
; Subroutine per .....
; input : registri, significato e valori possibili oppure void
; output: registri, significato e valori possibili oppure void
NomeSub PROC
      utilizza i valori ricevuti dal programma chiamante (se necessario)
      :::::::::::
      calcola il valore della risposta nei registri di output (se necessario)
      ret
NomeSub ENDP

```

Esempi:

```

;-----
; Sub che continua a leggere la tastiera finché non sia battuto il tasto 'C'.
; input : void
; output: void
LeggiTastoC PROC
      push Ax
LeggiTastoCciclo:
      mov  Ah, 0
      int 0x16
      cmp  Al, 67
      jne LeggiTastoCciclo:
      pop  Ax
      ret
LeggiTastoC ENDP
;-----
; Sub che continua a leggere la tastiera finché non sia battuto
; il carattere indicato dal programma chiamante.
; input : CL = carattere ASCII da accettare dalla tastiera
; output: void
LeggiTasto PROC
      push Ax
LeggiTastociclo:
      mov  Ah, 0 ; servizio di lettura dell'interrupt di tastiera
      int 0x16
      cmp  Al, Cl
      jne LeggiTastociclo:
      pop  Ax
      ret
LeggiTasto ENDP
;-----
; Sub che riceve un numero intero (word) e restituisce
; 1 se il numero è dispari, 0 se il numero è pari.
; input : AX = numero da esaminare
; output: BX = 1 se AX pari, 0 se AX dispari
PariDispari PROC
      xor  Bx, Bx
      test Ax, 1
      jz  PariDispariEsci
      inc Bx
PariDispariEsci:
      ret
PariDispari ENDP

```

```

;-----
; Sub che riceve un numero intero (word) e restituisce
; 1 se il numero è negativo, 0 se il numero è positivo.
; input  : AX = numero da esaminare
; output : BX = 1 se AX negativo, 0 se AX positivo
Segno PROC
    xor  Bx, Bx
    cmp  Ax, 0          |      test Ax, 0x8000
    jge  SegnoEsci     |      jz   SegnoEsci
    inc  Bx
SegnoEsci:
    ret
Segno ENDP

;-----
; Sub che riceve un carattere ASCII e restituisce 1 se = '2' o '4' o '7'; viceversa 0.
; input  : CL = carattere ASCII ricevuto
; output : CH = 1 se CL = 50, 52 o 55; viceversa 0
Cerca247 PROC
    mov  Ch, 1          ; ipotizziamo il vero
    cmp  Cl, 50         ; carattere '2'
    je   Cerca247Esci
    cmp  Cl, 52         ; carattere '4'
    je   Cerca247Esci
    cmp  Cl, 55         ; carattere '7'
    je   Cerca247Esci
    mov  Ch, 0          ; fallite le ricerche, azzeriamo Ch
Cerca247Esci:
    ret
Cerca247 ENDP

;-----
; Sub che renda NERO lo sfondo del primo rigo dello schermo.
; input  :
; output :
RigoNero PROC
    push Es
    push Ax
    push Bx
    mov  Ax, 0xB800
    mov  Es, Ax
    mov  Bx, 158        ; ultima casella del primo rigo
RigoNeroCiclo:
    mov  Ax, Es:[Bx]
    and  Ah, 0x0F
    mov  Es:[Bx], Ax
    sub  Bx, 2
    jns  RigoNeroCiclo

    pop  Bx
    pop  Ax
    pop  Es
    ret
RigoNero ENDP

;-----
; Sub che riceve un numero intero e lo raddoppia ripetutamente finché non diventi
; maggiore di 10000; al termine restituisce il valore ottenuto.
; input  : AX = numero da esaminare
; output : BX = valore maggiore di 10000 ottenuto
Sopra10K PROC
    mov  Bx, Ax
Sopra10KCiclo:
    cmp  Bx, 10000
    ja   Sopra10KEsci
    shl  Bx, 1          |      add Bx, Bx
    jmp  Sopra10KCiclo
Sopra10KEsci:
    ret
Sopra10K ENDP

```

```

;-----
; Sub che riceve un numero intero e lo dimezza ripetutamente finché non
; diventi minore di 100; al termine restituisce il valore ottenuto.
; input  : AX = numero da esaminare
; output : BX = valore minore di 100 ottenuto
Sotto100 PROC
    mov  Bx, Ax
Sotto100Ciclo:
    cmp  Bx, 100
    jb   Sotto100Esci
    shr  Bx, 1          ; dimezza il valore
    jmp  Sotto100Ciclo
Sotto100Esci:
    ret
Sotto100 ENDP

;-----
; Sub che riceve un intero N e restituisce 1 se N è multiplo di 7, viceversa 0.
; input  : AX = numero da esaminare
; output : BX = 1 se AX multiplo di 7, 0 viceversa
Multiplo7 PROC
    push Ax
    push Cx
    push Dx
    xor  Bx, Bx        ; supponiamo Ax non sia multiplo di 7
    mov  Cx, 7
    div  Cx             ; divide  DxAx / Cx  -> resto in Dx  quoto in Ax
    cmp  Dx, 0
    jne  Multiplo7Esci ; ipotesi iniziale corretta
    inc  Bx
Multiplo7Esci:
    pop  Dx
    pop  Cx
    pop  Ax
    ret
Multiplo7 ENDP

;-----
; Sub che legge un tasto e restituisce il corrispondente valore (da 0 a 15)
; se è una cifra esadecimale (0123456789ABCDEF), viceversa -1.
; input  : void
; output : CX = valore da 0 a 15 per cifre Hex; viceversa -1
LeggiHex PROC
    push Ax
    mov  Cx, 0xFFFF    ; -1 : ipotizziamo lettura errata
    mov  Ah, 0          ; servizio di lettura dell'interrupt di tastiera
    int  0x16
    cmp  Al, 48
    jb   LeggiHexEsci
    cmp  Al, 57
    ja   LeggiHexAvanti
    sub  Al, 48         ; se è una cifra da '0' a '9'
    xor  Ah, Ah
    mov  Cx, Ax
    jmp  LeggiHexEsci

LeggiHexAvanti:
    cmp  Al, 65
    jb   LeggiHexEsci
    cmp  Al, 70
    jb   LeggiHexEsci
    sub  Al, 55         ; se è una cifra da 'A' a 'F'
    xor  Ah, Ah
    mov  Cx, Ax

LeggiHexEsci:
    pop  Ax
    ret
LeggiHex ENDP

```

```

;-----
; Sub che somma tutti gli interi 16 bit delle celle da 8A4B0 a 8A4DE.
; input  : void
; output : BX = somma dei valori ottenuta
SommaCelle PROC
    push Es
    push Si
    mov  Bx, 0x8A40      ; indirizzo Base 8A400
    mov  Es, Bx
    mov  Si, 0x00B0
    xor  Bx, Bx          ; azzera il contenitore della somma
SommaCelleCiclo:
    add  Bx, Es:[Si]
    add  Si, 2           ; numero successivo
    cmp  Si, 0x00DE     ; cella finale
    jbe  SommaCelleCiclo
    pop  Si
    pop  Es
    ret
SommaCelle ENDP

;-----
; Sub che riceve un intero N e calcola la somma dei numeri pari minori o uguali a N.
; input  : AX = numero limite
; output : BX = somma dei numeri pari ottenuta
SommaPari PROC
    push Cx
    xor  Bx, Bx          ; azzera il contenitore della somma
    xor  Cx, Cx          ; numeri pari = 0, 2, 4, . . .
SommaPariCiclo:
    add  Bx, Cx
    add  Cx, 2           ; numero pari successivo
    cmp  Cx, Ax
    jbe  SommaPariCiclo
    pop  Cx
    ret
SommaPari ENDP

;-----

```

Sviluppare gli altri esempi :

Scrivere una Sub che :

- o colori di GIALLO i caratteri del primo rigo dello schermo.
- o renda VERDE lo sfondo del terzo rigo dello schermo.
- o renda BLU lo sfondo della prima colonna dello schermo.
- o colori di ROSSO i caratteri della prima colonna dello schermo.
- o riceve un carattere ASCII e restituisce 1 se è una lettera minuscola, 0 viceversa.
- o riceve un carattere ASCII e restituisce 1 se è una lettera MAIUSCOLA, 0 viceversa.
- o riceve un carattere ASCII e restituisce 1 se è una cifra numerica, viceversa 0.
- o stampa una 'A' alla riga e alla colonna indicata dal programma chiamante.
- o colori di Rosso/sfondo Blu la casella alla riga e alla colonna indicata dal programma chiamante.
- o conta il numero di '@' presenti sullo schermo.
- o conta i caratteri ROSSI presenti sullo schermo.
- o riceve un numero intero (word) e restituisce 0 se il numero è minore di 1000, viceversa 1.
- o riceve un numero intero e restituisce il numero dimezzato se è maggiore di 100, viceversa 1.
- o ordina i valori delle celle RAM 8A4B6 e 8A4B8 (scambiandoli se il primo è maggiore).