

Politecnico di Torino - III Facoltà di Ingegneria
Esercitazioni del corso “Calcolatori Elettronici 2”
Ing. Paolo Bernardi – email: paolo.bernardi@polito.it

Esercitazione 5 – L’acquisizione di dati da tastiera e la visualizzazione di dati a video.

Argomenti trattati:

1. L’istruzione INT.
2. L’acquisizione di dati da tastiera.
3. La visualizzazione di dati a video.

Esercizio 1: Scrivere un programma Assembler che legga da tastiera un carattere; se questo ha codice ASCII maggiore di 127, il programma deve sostituirlo con il carattere asterisco (*). Il programma deve scrivere nel bit più significativo del carattere ottenuto (che sicuramente è a 0) la “parità” del carattere (0 se il numero di bit uguali a 1 è pari, 1 in caso contrario), invertirlo e porre il risultato nella locazione di memoria SEND.

Possibile svolgimento:

```
LF          EQU    10
CR          EQU    13
STAR       EQU    '*'
           .MODEL SMALL
           .STACK
           .DATA
SEND       DB     (?)
           .CODE
           .STARTUP
next:      MOV     AH,    1
           INT     21H
           MOV     BL,    AL
           MOV     AH,    2
           MOV     DL,    LF
           INT     21H
           MOV     DL,    CR
           INT     21H
           TEST    BL,    80H
           JZ     minore
minore:    MOV     BL,    STAR
           MOV     CX,    8
           MOV     DH,    0
           MOV     BH,    BL
prox:      SHR     BH,    1
           JNC    zero
           INC    DH
zero:      LOOP   prox
           TEST   DH,    01H
           JZ     pari
           OR     BL,    80H
pari:      MOV     BH,    0
           MOV     CX,    7
invert:    SHL     BL,    1
           JNC    inzero
           OR     BH,    80H
inzero:    SHR     BH,    1
           LOOP   invert
           OR     BH,    BL
           MOV     SEND,  BH
           .EXIT
END
```

Esercizio 2: Scrivere un programma Assembler che stampi su video una stringa per la richiesta di introduzione di un numero intero, legga il numero e stampi su video un messaggio che specifichi se il numero che è stato introdotto è pari o dispari.

Possibile svolgimento:

```

LF          EQU    10
CR          EQU    13
           .MODEL SMALL
           .STACK
           .DATA
PROMPT     DB      'Inserisci un numero: ','$'
PARI       DB      "Il numero che hai scritto e' pari.",'$'
DISPARI    DB      "Il numero che hai scritto e' dispari.","$"
           .CODE
           .STARTUP ; acquisisco un numero
LEA DX,    PROMPT
MOV AH,    9
INT 21H
MOV AH,    1
INT 21H
MOV BL,    AL
MOV AH,    2
MOV DL,    LF
INT 21H
MOV DL,    CR
INT 21H
SUB BL,    '0'
TEST BL,   1
JZ  par
LEA DX,    DISPARI
MOV AH,    9
INT 21H
JMP fine
par:       LEA DX,    PARI
           MOV AH,    9
           INT 21H
fine:      MOV AH,    2
           MOV DL,    LF
           INT 21H
           MOV DL,    CR
           INT 21H
           .EXIT
           END

```

Esercizio 3: Scrivere un programma Assembler che legga da tastiera un numero compreso fra 0 e 9, moltiplichi per tale numero un vettore di 5 numeri positivi su due cifre (cioè compresi fra 00 e 99), e stampi su video il massimo degli elementi del vettore così ottenuto purché sia maggiore di 100; in caso contrario il programma deve stampare il carattere '#'.

Possibile svolgimento:

```

DIM        EQU    5
LF         EQU    10
CR         EQU    13
MAXVAL     EQU    100
           .MODEL SMALL
           .STACK
           .DATA
VETT       DB      DIM DUP (?)

```

```

.CODE
.STARTUP
MOV  AH,  1          ; preparo AH per la lettura
INT  21H           ; leggo il fattore moltiplicativo, lo metto
                    ; in AL
MOV  BL,  AL        ; e poi lo salvo in BL
MOV  AH,  2          ; preparo AH per la scrittura
MOV  DL,  LF        ; vado alla prossima riga
INT  21H
MOV  DL,  CR        ; vado a capo
INT  21H
CMP  BL,  '0'       ; se BL < '0'
JB   errinp        ; non e' una cifra, salto a errinp
CMP  BL,  '9'       ; se BL > '9'
JA   errinp        ; non e' una cifra, salto a errinp
SUB  BL,  '0'       ; converti BL in cifra
next: MOV  SI,  0      ; SI punta alla casella corrente di VETT
MOV  AH,  1          ; preparo AH per la lettura
INT  21H           ; leggo la prima cifra
SUB  AL,  '0'       ; converto in binario
MOV  BH,  10        ; preparo BH per la moltiplicazione
MUL  BH            ; in AX ora c'e' BH*AL
MOV  VETT[SI], AL   ; metto AL in VETT[SI] (decine), AH tutti
                    ; zeri
MOV  AH,  1          ; preparo AH per la lettura
INT  21H           ; leggo la seconda cifra
SUB  AL,  '0'       ; converto
ADD  VETT[SI], AL   ; completo la conversione
MOV  AH,  2          ; preparo AH per la scrittura
MOV  DL,  LF        ; vado alla prossima riga
INT  21H
MOV  DL,  CR        ; vado a capo
INT  21H
INC  SI            ; incremento SI
CMP  SI,  DIM      ; ho caricato tutto il vettore ?
JNE  next         ; se no, salto a next
MOV  CX,  0        ; CX contiene il massimo corrente
MOV  SI,  0        ; SI punta alla casella corrente
prox: MOV  AL,  VETT[SI] ; elemento da moltiplicare
MUL  BL           ; moltiplico AL*BL, risultato in AX
CMP  CX,  AX       ; confronto AX con il massimo corrente
JAE  nosc         ; non scambio, salto a nosc
XCHG AX,  CX       ; scambio AX e CX
nosc: INC  SI       ; incremento SI
CMP  SI,  DIM      ; ho esplorato tutto il vettore ?
JNE  prox         ; se no, salto a prox
CMP  CX,  MAXVAL   ; massimo > di MAXVAL ?
JB   minore       ; se no, salto a minore
MOV  AX,  CX       ; metto il massimo in AX
MOV  BH,  100      ; preparo BH per la divisione
DIV  BH
MOV  DL,  AL        ; metto le centinaia in DL
ADD  DL,  '0'       ; converto in ASCII le centinaia
MOV  BL,  AH        ; sposto il resto in BL
MOV  AH,  2          ; preparo AH per la stampa
INT  21H           ; stampo le centinaia
MOV  AL,  BL        ; metto il resto in AL
CBW                    ; trasformo AL in AX
MOV  BH,  10        ; preparo BH per la divisione
DIV  BH
MOV  DL,  AL        ; metto le decine in DL
ADD  DL,  '0'       ; converto in ASCII le decine
MOV  BL,  AH        ; sposto il resto in BL

```

```

                MOV     AH,    2           ; preparo AH per la stampa
                INT     21H           ; stampo le decine
                MOV     DL,    BL       ; metto il resto in DL
                ADD     DL,    '0'      ; converto in ASCII le unita'
                INT     21H           ; stampo le unita'
                JMP     fine
errinp:        MOV     AH,    2           ; preparo AH per la stampa
                MOV     DL,    '*'      ; metto '*' in DL
                INT     21H ; stampo
                JMP     fine
minore:       MOV     AH,    2           ; preparo AH per la stampa
                MOV     DL,    '#'      ; metto '#' in DL
                INT     21H           ; stampo
fine:         MOV     DL,    LF        ; vado sulla prossima riga
                INT     21H
                MOV     DL,    CR        ; vado a capo
                INT     21H
                .EXIT
                END

```