Soluzioni esercizi per laboratorio Assembly – 27/4/2016

Si legga da tastiera con la funzione 00h dell'interrupt 16h una sequenza di numeri. Il programma deve verificare se sono cifre da 0 a 9 (e se non lo sono, non considerarle) fino alla pressione del tasto ESC. Il numero risultante deve essere convertito in un numero intero e memorizzato in una variabile Numero. Esempio: se l'utente inserisce:
1f3rt 34
In Numero deve essere memorizzato il numero 1334.

```
SECTION data
Numero      DW 0 ; suppongo che per il numero inserito bastino 16 bit altrimenti dovrei usare un DD
doubleword

SECTION text

..start:
Leggi:      mov ax, data
            mov ds,ax                   ; faccio puntare DS alle variabili
            xor bx,bx                   ; uso bx per contare il numero di cifre corrette inserite
Ripeti:     mov     ah,00h              ; Questa è la funzione di lettura
            int     16h                         ; di un carattere
            cmp     al,1bh              ; Il codice ASCII è 1B (ESC)?
            je Fine             ; Se sì, vado a Converti

            cmp al,'0'
            jb Ripeti ; se è minore di '0' non può essere una cifra; torno a Ripeti e ne chiedo un'altra
            cmp al, '9'
            ja Ripeti ; se è maggiore di '9' non può essere una cifra; torno a Ripeti e ne chiedo un'altra
            mov bl, al
            sub bl, 30h             ; converto il codice ascii del numero nel numero vero e proprio
            mov ax, [Numero]
            mov dx, 10
            mul dx  ; il risultato in DX:AX, ma per l'ipotesi fatta all'inizio non mi interessa DX, ma solo
AX
            add ax, bx                  ; vecchio numero per 10 + nuovo numero (in BX)
            mov [Numero], ax
            jmp Ripeti
Fine:       mov ax, 4C00h           ; servizio esci (return code=0)
            int 21h
```

Si legga da tastiera con la funzione 00h dell'interrupt 16h una sequenza di caratteri e si visualizzino (con la funzione 0eh dell'interrupt 10h) solo i caratteri minuscoli (da 'a' a 'z'). Gli altri caratteri non devono essere visualizzati. La sequenza termina alla pressione del tasto ESC.

```
Leggi:  mov   ah,00h        ; Questa è la funzione di lettura
              int   16h           ; di un carattere

              cmp   al,1Bh        ; Il codice ASCII è 1B (ESC)?
              je    Fine          ; Se sì, vai alla fine

              cmp al, 'a'         ;
              jb Leggi            ; se minore di 'a' chiedo nuovo carattere
              cmp al, 'z'         ;
              ja Leggi            ; se maggiore di 'z' chiedo nuovo carattere

              mov   ah,0eh        ; Funzione di scrittura a video
              mov   bx,00h        ; Pagina 0 (BH)
              int   10h
              jmp   Leggi         ; Leggi un altro carattere

Fine:         MOV AX, 4C00h       ; servizio esci (return code=0)
              INT 21h
```

Si legga da tastiera con la funzione 00h dell'interrupt 16h una sequenza di caratteri e si visualizzino (con la funzione 0eh dell'interrupt 10h) i caratteri solo maiuscoli (trasformando le lettere da minuscole a maiuscole) e un * al posto di caratteri diversi dalle lettere. La sequenza termina alla pressione del tasto ESC.

```
Leggi:          mov   ah,00h        ; Questa è la funzione di lettura
                      int   16h           ; di un carattere

                      cmp   al,1Bh        ; Il codice ASCII è 1B (ESC)?
                      je    Fine          ; Se sì, vai alla fine

                      cmp al, 'a'
                      jb NonMinuscola
                      cmp al, 'z'
                      ja NonMinuscola
                      ; qui arrivo solo se il carattere è una minuscola
                      sub al, 20h     ; la distanza nella tabella ascii tra minuscola e maiuscola è
20h (es: 'A'=41h 'a'=61h)
                      jmp Stampa
NonMinuscola: cmp al, 'A'
                      jb NonLettera   ; qui so già che non è una lettera e la salto
                      cmp al, 'Z'
                      ja NonLettera   ; siccome ho già considerato il caso delle lettere minuscole
anche qui so che non è una lettera e la salto
                      ; qui arrivo solo se il carattere è una maiuscola e non devo fare nulla
                      jmp Stampa
NonLettera:     mov al, '*'

Stampa:               mov   ah,0eh        ; Funzione di scrittura a video
                      mov   bx,00h        ; Pagina 0 (BH)
```

```
                    int     10h
                    jmp     Leggi           ; Leggi un altro carattere

Fine:   MOV AX, 4C00h       ; servizio esci (return code=0)
        INT 21h
```

Si legga da tastiera con la funzione 00h dell'interrupt 16h una sequenza di caratteri e li si visualizzi sullo schermo con la funzione 0eh dell'interrupt 10h. Ogni volta che viene inserito uno spazio, si scriva sullo schermo (in una nuova linea) il numero di caratteri scritti prima dello spazio. La sequenza termina alla pressione del tasto ESC. Esempio: se l'utente inserisce
Buonasera a tutti
L'output sullo schermo deve essere:
Buonasera
9
a
1
tutti
5

```
SECTION data
Conta:          db 0

SECTION text

                    mov cl,[Conta]  ; variabile per contare il numero di caratteri di una parola
Leggi:              mov     ah,00h          ; Questa è la funzione di lettura
                    int     16h                     ; di un carattere

                    cmp     al,1Bh          ; Il codice ASCII è 1B (ESC)?
                    je      Fine            ; Se sì, vai alla fine

                    cmp al, 20h             ; codice ascii dello spazio (20h o 32 decimale)
                    je Spazio
                    inc cl                  ; conto i caratteri diversi da spazio
                    call Stampa
                    jmp     Leggi           ; Leggi un altro carattere
Spazio:             call ACapo
        call TrasformaNumero ; trasforma quanto contenuto in BL nei caratteri
; ascii corrispondenti e li stampa
                    jmp Leggi

Fine:               mov ax, 4C00h       ; servizio esci (return code=0)
                    int 21h

Stampa:             mov     ah,0eh          ; Funzione di scrittura a video
                    mov     bx,00h          ; Pagina 0 (BH)
                    int     10h
                    ret

TrasformaNumero:
                    ; supponiamo il numero inferiore a 1000 quindi composto da massimo 3
cifre numeriche
```

```
                    mov [Conta], cl
                    mov al, cl
                    xor ah, ah
                    mov dl, 100
                    div dl              ; quoziente in AL, resto in AH
                    push ax             ; salvo il risultato sullo stack
                    cmp al, 0
                    je Salto1
                    add al, 30h         ; trasformo il numero nel suo codice ascii
                    call Stampa
Salto1:             pop ax              ; recupero dallo stack
                    xchg al, ah         ; ora in al ho il resto della divisione precedente
                    xor ah, ah
                    mov dl, 10
                    div dl
                    push ax             ; salvo il risultato sullo stack
                    cmp al, 0
                    je Salto2
                    add al, 30h
                    call Stampa
Salto2:             pop ax              ; recupero dallo stack
                    xchg al, ah
                    add al, 30h         ; trasformo il numero nel suo codice ascii
                    call Stampa
                    call ACapo
                    mov cl, 0 ; riazzero il contatore
                    ret

ACapo:              mov al, 0dh         ; Carriage return (CR)
                    call Stampa
                    mov al, 0ah         ; Line Feed (LF)
; CR+LF sono equivalenti ad andare a capo (tasto ENTER)
                    call Stampa
                    ret
```

Sia memorizzata in una variabile una stringa stile Pascal (si vedano i lucidi a tal proposito). Il programma deve scorrere la stringa e contare le vocali, mettendo il risultato in una variabile e/o visualizzando il valore sullo schermo.
Rifare l'esercizio con una stringa zero terminata alla C.

VERSIONE CON STRINGHE PASCAL

```
SECTION data
Stringa:        db 18,'Buongiorno a tutti'
NrVocali:       resb 1

SECTION text
..start:
                mov ax, data
                mov ds, ax
                xor ch, ch
                mov cl, [Stringa]       ; metto in cl la lunghezza della stringa
                mov dl, 00h             ; uso dl per contare le vocali
                mov si, Stringa+1       ; si punta al primo carattere della stringa
Ciclo:          lodsb
                cmp al, 'a'             ; ipotizziamo solo minuscole.
                je Vocale               ; Se si vogliono considerare anche le
maiuscola basta aggiungere casi
                cmp al, 'e'
                je Vocale
                cmp al, 'o'
                je Vocale
                cmp al, 'i'
                je Vocale
                cmp al, 'u'
                je Vocale
                loop Ciclo
                jmp Visualizza
Vocale:         inc dl
                loop Ciclo
Visualizza:     mov [NrVocali], dl
                Call StampaNumero

                mov ax, 4C00h       ; servizio esci (return code=0)
                int 21h

StampaNumero:
                ; supponiamo il numero inferiore a 1000 quindi composto da massimo 3
cifre numeriche
                ; simile al precedente
                mov al, [NrVocali]
                xor ah, ah
                mov dl, 100
                div dl              ; quoziente in AL, resto in AH
                push ax             ; salvo il risultato sullo stack
                cmp al, 0
                je Salto1
                add al, 30h         ; trasformo il numero nel suo codice ascii
                call Stampa
```

```
Salto1:     pop ax              ; recupero dallo stack
            xchg al, ah         ; ora in al ho il resto della divisione precedente
            xor ah, ah
            mov dl, 10
            div dl
            push ax             ; salvo il risultato sullo stack
            cmp al, 0
            je Salto2
            add al, 30h
            call Stampa
Salto2:     pop ax              ; recupero dallo stack
            xchg al, ah
            add al, 30h         ; trasformo il numero nel suo codice ascii
            call Stampa
            ret

Stampa:     mov   ah,0eh        ; Funzione di scrittura a video
            mov   bx,00h        ; Pagina 0 (BH)
            int   10h
            ret
```

VERSIONE CON STRINGHE C (in corsivo le differenze)

```
SECTION data
Stringa:    db 'Buongiorno a tutti',0
NrVocali:   resb 1

SECTION text
..start:
            mov ax, data
            mov ds, ax
            mov dl, 00h             ; uso dl per contare le vocali
            mov si, Stringa         ; si punta al primo carattere della stringa
Ciclo:      lodsb
            cmp al, 0               ; fine stringa (terminatore C)
            je Visualizza
            cmp al, 'a'             ; ipotizziamo solo minuscole.
            je Vocale               ; Se si vogliono considerare anche le
; maiuscola basta aggiungere casi
            cmp al, 'e'
            je Vocale
            cmp al, 'o'
            je Vocale
            cmp al, 'i'
            je Vocale
            cmp al, 'u'
            je Vocale
            jmp Ciclo
Vocale:     inc dl
            jmp Ciclo
Visualizza: mov [NrVocali], dl
            Call StampaNumero

            mov ax, 4C00h       ; servizio esci (return code=0)
```

```
                    int 21h

StampaNumero:
                    ; supponiamo il numero inferiore a 1000 quindi composto da massimo 3
cifre numeriche
                    ; simile al precedente
                    mov al, [NrVocali]
                    xor ah, ah
                    mov dl, 100
                    div dl              ; quoziente in AL, resto in AH
                    push ax             ; salvo il risultato sullo stack
                    cmp al, 0
                    je Salto1
                    add al, 30h         ; trasformo il numero nel suo codice ascii
                    call Stampa
Salto1:     pop ax                  ; recupero dallo stack
                    xchg al, ah         ; ora in al ho il resto della divisione precedente
                    xor ah, ah
                    mov dl, 10
                    div dl
                    push ax             ; salvo il risultato sullo stack
                    cmp al, 0
                    je Salto2
                    add al, 30h
                    call Stampa
Salto2:     pop ax                  ; recupero dallo stack
                    xchg al, ah
                    add al, 30h         ; trasformo il numero nel suo codice ascii
                    call Stampa
                    ret

Stampa:             mov     ah,0eh      ; Funzione di scrittura a video
                    mov     bx,00h      ; Pagina 0 (BH)
                    int     10h
                    ret
```

Sia memorizzata in una variabile una stringa stile Pascal (si vedano i lucidi a tal proposito). Il programma deve scorrere la stringa e riscriverla in una nuova variabile stringa in ordine inverso. Si visualizzi la nuova stringa sullo schermo.
Rifare l'esercizio con una stringa zero terminata alla C.

VERSIONE CON STRINGHE PASCAL

```
SECTION data
Stringa1:           db 18,'Buongiorno a tutti'
Stringa2:           resb 256

SECTION text
..start:
                    mov ax, data
                    mov ds, ax
                    mov es, ax
```

```
                    xor ch, ch
                    mov cl, [Stringa1]      ; metto in cl la lunghezza della stringa
                    mov [Stringa2], cl
                    mov si, Stringa1
                    add si, cx                  ; così SI punta all'ultimo carattere di
Stringa1
                    mov di, Stringa2+1
Ciclo:              std                         ; metto il direction flag DF a 1 in
modo che SI venga decrementato
                    lodsb
                    cld                         ; metto il direction flag DF a 0 in
modo che DI venga incrementato
                    stosb
                    loop Ciclo

                    ; Visualizzo Stringa2
                    mov     cl,[Stringa2]
                    mov     si,Stringa2+1
                    mov     ah,0eh
                    mov     bx,0000h
Stampa2:        lodsb
                    int     10h
                    loop    Stampa2


                    mov ax, 4C00h       ; servizio esci (return code=0)
                    int 21h
```

VERSIONE CON STRINGHE C

```
SECTION data
Stringa1:       db 'Buongiorno a tutti',0
Stringa2:       resb 256

SECTION text
..start:
                    mov ax, data
                    mov ds, ax
                    mov es, ax
                    xor cx, cx      ; uso contatore per i caratteri da copiare
                    mov si, Stringa1
                    ; devo scorrere la stringa fino a trovare il terminatore
ScorriStringa:  lodsb
                    cmp al, 0
                    je FineStringa
                    inc cx
                    jmp ScorriStringa
FineStringa:    dec si                  ; indietro di due per posizionarsi sull'ultimo carattere

                    dec si
                    mov di, Stringa2
Ciclo:          std                     ; metto il direction flag DF a 1 in modo che
SI venga decrementato
                    lodsb
```

```
                    cld                                 ; metto il direction flag DF a 0 in
modo che DI venga incrementato
                    stosb
                    loop Ciclo
FineCiclo:          mov [di],byte 0         ; metto il terminatore in Stringa2

                    ; Visualizzo Stringa2
                    cld
                    mov     si,Stringa2
                    mov     ah,0eh
                    mov     bx,0000h
Stampa2:            lodsb
                    cmp al, 0
                    je Fine
                    int     10h
                    jmp     Stampa2

Fine:               mov ax, 4C00h       ; servizio esci (return code=0)
                    int 21h
```

Siano memorizzate due variabili stringa stile Pascal (si vedano i lucidi a tal proposito). Il programma deve visualizzare sullo schermo un carattere di ognuna stringa in modo alternato.
Rifare l'esercizio con una stringa zero terminata alla C.

VERSIONE CON STRINGHE PASCAL

```
SECTION data
Stringa1:       db 18, 'Buongiorno a tutti'
Stringa2:       db 14, 'Bella giornata'

SECTION text
..start:
                mov ax, data
                mov ds, ax
                mov es, ax
                xor ch, ch
                mov cl, [Stringa1]
                xor dh, dh
                mov dl, [Stringa2]
                mov si, 0
                mov di, 0
Ciclo:          mov al, [Stringa1+1+si]
                call Stampa
                inc si
                cmp si, cx                      ; sono arrivato a fine della
stringa1?
                je FineStringa1
                mov al, [Stringa2+1+di]
                call Stampa
                inc di
                cmp di, dx                      ; sono arrivato a fine della
stringa2?
                je FineStringa2
                jmp Ciclo
FineStringa1:   ; finisco di copiare stringa2
                mov al, [Stringa2+1+di]
                call Stampa
                inc di
                cmp di, dx                      ; sono arrivato a fine della
stringa2?
                je FineTutto
                jmp FineStringa1
FineStringa2:   ; finisco di copiare stringa1
                mov al, [Stringa1+1+si]
                call Stampa
                inc si
                cmp si, cx                      ; sono arrivato a fine della
stringa2?
                jne FineStringa2
FineTutto:
                mov ax, 4C00h       ; servizio esci (return code=0)
                int 21h

Stampa:         mov     ah,0eh          ; Funzione di scrittura a video
```

```
                    mov     bx,00h          ; Pagina 0 (BH)
                    int     10h
                    ret
```

VERSIONE CON STRINGHE C (in corsivo le differenze)

```
SECTION data
Stringa1:           db 'Buongiorno a tutti',0
Stringa2:           db 'Bella giornata', 0

SECTION text
..start:
                    mov ax, data
                    mov ds, ax
                    mov es, ax
                    mov si, 0
                    mov di, 0
Ciclo:              mov al, [Stringa1+si]
                    cmp al, 0
                    je FineStringa1
                    call Stampa
                    inc si
                    mov al, [Stringa2+di]
                    cmp al, 0
                    je FineStringa2
                    call Stampa
                    inc di
                    jmp Ciclo
FineStringa1:   ; finisco di copiare stringa2
                    mov al, [Stringa2+di]
                    cmp al, 0
                    je FineTutto
                    call Stampa
                    inc di
                    jmp FineStringa1
FineStringa2:   ; finisco di copiare stringa1
                    mov al, [Stringa1+si]
                    cmp al, 0
                    je FineTutto
                    call Stampa
                    inc si
                    jmp FineStringa2
FineTutto:
                    mov ax, 4C00h       ; servizio esci (return code=0)
                    int 21h

Stampa:             mov     ah,0eh          ; Funzione di scrittura a video
                    mov     bx,00h          ; Pagina 0 (BH)
                    int     10h
                    ret
```

Siano memorizzate due variabili stringa stile Pascal (si vedano i lucidi a tal proposito). Il programma deve verificare quale stringa è "maggiore" dell'altra (stile la funzione strcmp del C) e visualizzare la stringa maggiore sullo schermo.
Rifare l'esercizio con una stringa zero terminata alla C.

VERSIONE CON STRINGHE PASCAL

```
SECTION data
Stringa1:       db 18, 'Buongiorno a tutti'
Stringa2:       db 15, 'Buongiorno a me'
Testo:          db 18, 'Stringhe identiche'

SECTION text
..start:
                mov ax, data
                mov ds, ax
                mov es, ax
                xor ch, ch
                mov cl, [Stringa1]
                cmp cl, [Stringa2]      ; verifico quale stringa è più corta
                jb PrimaPiuCorta
                mov cl, [Stringa2]
PrimaPiuCorta:  mov si, Stringa1+1
                mov di, Stringa2+1
repe            cmpsb
                cmp cx, 0               ; se sono uscito dall'istruzione precedente
con ZF=0
                                        ; allora le stringhe sono diverse e CX
non è arrivato
                                        ; a zero. Devo verificare quale è
minore
                je Identiche
                mov al, [si-1]          ; confronto i caratteri su cui mi sono fermato per
verificare il minore dei due
                cmp al, [di-1]          ; -1 perchè cmpsb ha comunque incrementato DI e
SI
                ja PrimaMaggiore
                ; altrimenti prima minore
                mov si, Stringa2+1  ; è la seconda la maggiore
                mov cl, [Stringa2]
                call ScriviStringa
                jmp Fine
PrimaMaggiore:  mov si, Stringa1+1
                mov cl, [Stringa1]
                call ScriviStringa
                jmp Fine
Identiche:      mov si, Testo+1
                mov cl, [Testo]
                call ScriviStringa
Fine:
                mov ax, 4C00h       ; servizio esci (return code=0)
                int 21h

ScriviStringa:
```

```
                    mov     ah,0eh          ; Funzione di scrittura a video
                    mov     bx,00h          ; Pagina 0 (BH)
Ciclo:      lodsb
                    int     10h
                    loop Ciclo
                    ret
```

VERSIONE CON STRINGHE C

```
SECTION data
Stringa1:           db 'Buongiorno a tutti',0
Stringa2:           db 'Buongiorno a me', 0
Testo:              db 'Stringhe identiche', 0

SECTION text
..start:
                    mov ax, data
                    mov ds, ax
                    mov es, ax
                    mov si, Stringa1
                    mov di, Stringa2
Ciclo:      lodsb                   ; carico in al da DS:SI quindi Stringa1
                    scasb                   ; confronto al con ES:DI quindi Stringa2
                    ; NOTA: non serve verificare il fine stringa. Il terminatore 0 viene trattato
come qualsiasi altro carattere
                    ja PrimaMaggiore
                    jb PrimaMinore
                    cmp al, 0               ; se arrivo qui vuol dire che le due stringhe
sono finora uguali.
                                            ; Se ho il terminatore le stringhe
sono identiche
                    je Identiche
                    jmp Ciclo
PrimaMaggiore:
                    mov si, Stringa1
                    call ScriviStringa
                    jmp Fine
PrimaMinore:
                    mov si, Stringa2
                    call ScriviStringa
                    jmp Fine
Identiche:
                    mov si, Testo
                    call ScriviStringa
Fine:
                    mov ax, 4C00h       ; servizio esci (return code=0)
                    int 21h

ScriviStringa:
                    mov     ah,0eh          ; Funzione di scrittura a video
                    mov     bx,00h          ; Pagina 0 (BH)
Ciclo2:     lodsb
                    cmp al, 0
                    je FineFunzione
```

```
                    int     10h
                    jmp Ciclo2
FineFunzione:  ret
```

Scrivere in Assembler per Intel 80x86 un programma che avendo un dato di tipo array di N interi ordinati (terminati dal valore -1) tipo:
vettoreOrdinato: dw -10, -8, -3, 0, 1, 10, 24, 33, 37, -1
elimini tutti gli elementi con valore N (inserito come ulteriore variabile) dal vettore. Il vettore è ordinato con valori crescenti di interi ed è terminato con il valore -1 (che non può essere mai presente come valore di un elemento).
Il programma deve rimuovere dal vettore ordinato tutte le occorrenze del valore N, mantenendo il vettore ordinato.

```
SECTION data
vettoreOrdinato: dw -10, -8, -3, 0, 1, 1, 1, 10, 24, 33, 37, -1
N:                  dw 1

SECTION text
..start:
                    mov ax, data
                    mov ds, ax
                    mov es, ax
                    mov si, vettoreOrdinato
Ciclo:          lodsw
                    cmp ax, -1
                    je Fine
                    cmp ax, [N]
                    jne Ciclo
                    push si                 ; devo salvare SI perchè Rimuovi lo modifica
                    call Rimuovi        ; in SI-1 ho l'elemento da rimuovere
                    pop si                  ; recupero SI
                    dec si                  ; riporto indietro SI di 2 perchè ho spostato
gli elementi del vettore indietro
                    dec si
                    jmp Ciclo
Fine:
                    mov ax, 4C00h   ; servizio esci (return code=0)
                    int 21h

Rimuovi:        mov di, si              ; elemento successivo a quello da rimuovere
                    dec di                  ; DI punta sull'elemento da rimuovere
                    dec di                  ; due decrementi per tenere conto che sono
word (2 byte)
                    ; copio tutti gli elementi da DS:SI a ES:DI
Ciclo2:         lodsw
                    stosw
                    cmp ax, -1              ; verifico se sono a fine vettore
                    je FineFunzione
                    jmp Ciclo2
FineFunzione:  ret
```

Scrivere in Assembler per Intel 80x86 un programma che ha una variabile di tipo array di interi. Il vettore non è ordinato e contiene valori byte positivi ed è terminato con il valore -1 (che non può essere mai presente come valore di un elemento). Il programma deve contare il numero di elementi dispari presenti nel vettore e metterlo in una ulteriore variabile.

```
SECTION data
vettore:        db 5, 4, 22, 12, 17, 3, 1, 8, -1
Ndispari:       db 0

SECTION text
..start:
                mov ax, data
                mov ds, ax
                mov si, vettore
Ciclo:          lodsb
                cmp al, -1
                je Fine
                test al, 1              ; il test è come l'and ma senza modifica del risultato
                                        ; farlo con il valore 1 (in binario 0000....01)
                                        ; dà valore 0 se il numero è pari, 1 se è dispari
                je Ciclo                ; numero pari
                inc byte [Ndispari]
                jmp Ciclo
Fine:
                mov ax, 4C00h           ; servizio esci (return code=0)
                int 21h
```