

**SOLUZIONI DELLA SECONDA PROVA INTERMEDIA DEL CORSO DI  
CALCOLATORI ELETTRONICI  
NUOVO ORDINAMENTO DIDATTICO**

7 Giugno 2003

**MOTIVARE IN MANIERA CHIARA LE SOLUZIONI PROPOSTE A CIASCUNO DEGLI ESERCIZI SVOLTI**

**ESERCIZIO 1 (8 punti)**

Considerato un campo di 36 bit, siano dati i seguenti formati:

1. rappresentazione di interi senza segno;
  2. rappresentazione in virgola fissa con 10 bit di parte frazionaria;
  3. rappresentazione in virgola mobile con mantissa frazionaria e normalizzata in segno e valore (0.M) ed esponente a 8 bit in eccesso 127.
- a) (2 punti) Calcolare il minimo e il massimo valore rappresentabile in valore assoluto nei tre casi.
- b) (2 punti) Rappresentare i valori  $(121.5)_{10}$   $(50.25)_{10}$  secondo la rappresentazione 3.
- c) (4 punti) Sommare i due numeri al punto precedente con l'algoritmo dei calcolatori.

**Soluzione**

a)

1. Minimo: 1 Max:  $2^{36}-1$ .

2. Minimo:  $2^{-10}$  Max:  $2^{25}-2^{-10}$

3. Minimo:  $2^{-128}$  Max:  $2^{128}(1-2^{-27})$ .

b)  $(121.5)_{10} = 1111001.1 = 0.11110011 * 2^7$   
 $(50.25)_{10} = 110010.01 = 0.11001001 * 2^6$

*I due numeri si possono rappresentare nel seguente modo:*

Segno Esponente Mantissa

0      10000110   1111001100000000000000000000

0      10000101   1100100100000000000000000000

c) Poiché il primo ha esponente maggiore del secondo ( $7 > 6$ ) di quest'ultimo si fa scorrere la mantissa a destra di una posizione.

*I due numeri da sommare sono:*

$$\begin{array}{r} 0.111100110 + \\ 0.011001001 = \\ \hline 1.010101111 \quad (*2^7) \end{array}$$

*E' necessario normalizzare il risultato*

Segno Esponente Mantissa

0      1000111   1010101111000000000000000000

## ESERCIZIO 2 (9 punti)

1) (8 punti) Scrivere una funzione Assembler MIPS che, ricevendo in ingresso un intero  $x$ , restituisca il numero di elementi presenti in un vettore  $v$  che risultano maggiori di  $x$ . La dimensione di  $v$  è di 100 elementi.

Si assuma che:  $\&v[0] \rightarrow$  locazione 1000,  $100 \rightarrow \$2$ ,  $x \rightarrow \$4$ , , valore di uscita  $\rightarrow \$5$ .

2) (1 punto) Si scriva il frammento di codice del chiamante che invoca detta funzione e poi scarica il risultato nella locazione di memoria 2000.

Es. Il codice Assembler MIPS potrebbe implementare il seguente codice C:

```
int maggiore(int x)
{
    int i, magg;
    magg=0;
    for(i=0; i<100; i++)
        if(v[i]>x) magg++;
    return magg;
}
```

### Soluzione.

Proponiamo una soluzione che fa uso dei seguenti registri per i dati intermedi:

$\$8 \leftarrow i$ ;  $\$9 \leftarrow i*4$ ;  $\$10 \leftarrow v[i]$ ;  $\$11 \leftarrow (v[i]>x)$

```
maggiore:    addi $29, $29, -16 #salvataggio contesto
             sw $8, 0($29)
             sw $9, 4($29)
             sw $10, 8($29)
             sw $11, 12($29)
             move $5, $0        #magg=0
             move $8, $0        #i=0
             move $9, $0
For:         beq $8, $2, Exit    #if(i==100) Exit
             lw $10, 1000($9)   #prelevo v[i]
             addi $8, $8, 1     #i++
             addi $9, $9, 4     #offset prossima parola
             slt $11, $4, $10   # $11 ← x<v[i]
             beq $11, $0, For   #if(!($11)) For
             addi $5, $5, 1     #magg++
             j For
Exit:        lw $8, 0($29)      #ripristino del contesto
             lw $9, 4($29)
             lw $10, 8($29)
             lw $11, 12($29)
             addi $29, $29, 16
             jr $31            #ritorno al chiamante
```

Codice del chiamante richiesto:

```
...
jal maggiore
sw $5, 2000($0)
...
```

**ESERCIZIO 3 (8 punti)**

Si consideri un sistema operativo con gestione della memoria paginata. Ciascun processo ha una dimensione massima di 64 pagine. Le pagine hanno dimensione 1KB. Si abbia ad esempio la seguente situazione:

| Tabella delle Pagine Processo 0 |               |                 |              |
|---------------------------------|---------------|-----------------|--------------|
| Pagina Virtuale                 | Pagina Fisica | Bit di Validità | Solo lettura |
| 000000                          | 0000011       | 0               | 0            |
| 000001                          | 0010110       | 0               | 0            |
| 000010                          | 1001001       | 1               | 1            |
| 000011                          | 1001010       | 1               | 1            |
| 000100                          | 1010101       | 1               | 1            |
| 000101                          | 0011101       | 1               | 0            |
| 000110                          | 0111111       | 1               | 0            |
| 000111                          | 1011101       | 1               | 0            |
| 001000                          | 1010011       | 1               | 0            |
| 001001                          | 0001111       | 1               | 0            |
| 001010                          | 0011011       | 1               | 0            |
| 001011                          | 0100010       | 1               | 0            |

| Tabella delle Pagine Processo 1 |               |                 |              |
|---------------------------------|---------------|-----------------|--------------|
| Pagina Virtuale                 | Pagina Fisica | Bit di Validità | Solo lettura |
| 000000                          | 0001010       | 1               | 0            |
| 000001                          | 0010001       | 0               | 0            |
| 000010                          | 1011001       | 1               | 0            |
| 000011                          | 1011010       | 1               | 0            |
| 000100                          | 0011101       | 1               | 0            |
| 000101                          | 0101111       | 0               | 0            |
| 000110                          | 0110111       | 1               | 0            |
| 000111                          | 0100000       | 1               | 0            |
| 001000                          | 0100100       | 1               | 0            |
| 001001                          | 0001001       | 1               | 0            |

| Tabella delle Pagine Processo 2 |               |                 |              |
|---------------------------------|---------------|-----------------|--------------|
| Pagina Virtuale                 | Pagina Fisica | Bit di Validità | Solo lettura |
| 000000                          | 0100001       | 1               | 1            |
| 000001                          | 0101110       | 1               | 1            |
| 000010                          | 0110110       | 0               | 1            |
| 000011                          | 0000110       | 1               | 1            |
| 000100                          | 1100011       | 0               | 1            |
| 000101                          | 1000011       | 1               | 1            |
| 000110                          | 0010101       | 1               | 1            |

Il bit di validità indica se la pagina virtuale richiesta è presente (1) o no (0) nella memoria principale.

1- Calcolare la dimensione in bit degli indirizzi virtuali, spiegando come vengono interpretati dal sistema operativo. (2 punti)

2- (4 punti) Si considerino le seguenti richieste e mostrare: a quale indirizzo fisico corrispondono gli indirizzi virtuali, se l'operazione può essere conclusa con successo o se viene generato un errore (pagina non valida, violazione di protezione) o un page fault:

1) (Processo 0) Read 0001000101001101

2) (Processo 2) Write 0000010101011001

3) (Processo 1) Read 0010001001110011

4) (Processo 0) Read 0001001100100110

5) (Processo 0) Write 0000110001100001

3- (2 punti) calcolare la dimensione massima di una PMT

---

**Soluzione:**

- 1) L'indirizzo virtuale ha dimensione pari almeno a 16 bit e viene interpretato a partire dai bit più significativi nel modo seguente:  
6 bit: pagina virtuale  
10 bit: posizione all'interno della pagina
- 2) A partire dalla interpretazione del punto 1. Si ottiene:  
(a) Indirizzo fisico valido: 1010101 0101001101  
(b) Errore: scrittura su indirizzo fisico 0101110 0101011001 protetto da scrittura  
(c) Indirizzo fisico valido: 0100100 1001110011  
(d) Indirizzo fisico valido: 1010101 1100100110  
(e) Errore: scrittura su indirizzo fisico 1001010 0001100001 protetto da scrittura
- 3) La PMT contiene al massimo 64 'righe', una per ciascuna pagina virtuale. Ciascuna riga contiene 6 bit per la pagina virtuale, 7 bit per la pagina fisica, 2 bit di controllo, per un totale di 15 bit. La dimensione massima di una PMT è dunque pari a  $64 \times 15 \text{ bit} = 120 \text{ byte}$ .

#### ESERCIZIO 4 (8 punti)

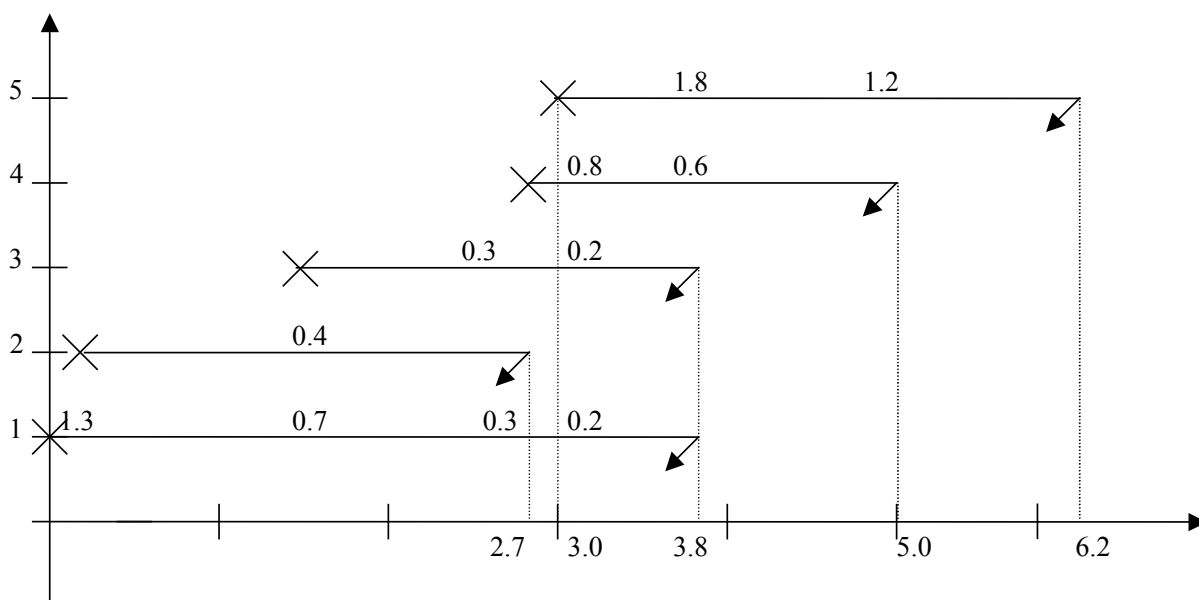
Sia data la seguente lista di processi:

| Processo | Tempo di arrivo | Tempo di CPU |
|----------|-----------------|--------------|
| 1        | 0.0             | 1.6          |
| 2        | 0.3             | 1.0          |
| 3        | 1.5             | 0.7          |
| 4        | 2.7             | 0.9          |
| 5        | 3.0             | 2.0          |

1. (6 punti) Mostrare, utilizzando il metodo grafico, la sequenza di esecuzione dei processi qualora si impieghi la politica di scheduling FIFO multiprogrammata.
2. Calcolare il tempo di *turnaround* medio e il tempo di *turnaround* pesato medio (2 punti).

#### Soluzione.

Grafico processo-tempo con politica FIFO multiprogrammata:



| Processo | Arrivo | Fine | Turnaround | WT  |
|----------|--------|------|------------|-----|
| 1        | 0.0    | 3.8  | 3.8        | 2.4 |
| 2        | 0.3    | 2.7  | 2.4        | 2.4 |
| 3        | 1.5    | 3.8  | 2.3        | 3.3 |
| 4        | 2.7    | 5.0  | 2.3        | 2.6 |
| 5        | 3.0    | 6.2  | 3.2        | 1.6 |
| Media    |        |      | 2.8        | 2.5 |