

- 1. L'algoritmo di Aging per la sostituzione delle pagine prevede di mantenere un contatore C associato ad ogni pagina caricata in memoria. Tale contatore viene consultato nel momento in cui si deve scegliere quale pagina rimuovere dalla memoria: viene scelta quella con il contatore più basso.**

**Indicare esattamente qual'è l'aggiornamento periodico che viene effettuato su tale contatore.**

- A. Somma del bit di referenziamento R al contatore C, con seguente shift a sinistra.
- B. Shift a sinistra di C e somma del bit di referenziamento R.
- C. Shift a sinistra di C ed inserimento del bit di referenziamento R come bit più significativo.
- D. Shift a destra di C ed inserimento del bit di referenziamento R come bit più significativo.
- E. Shift a destra di C ed inserimento del bit di modifica M come bit meno significativo.

- 2. Con riferimento alle tecniche che abbiamo visto per memorizzare il contenuto dei file sui blocchi del disco e di come il file-system ne tenga traccia, individuare quale tra le seguenti affermazioni è falsa.**

- A. Nell'allocazione contigua è necessario conoscere a priori la dimensione massima del file in fase di creazione.
- B. Nell'allocazione concatenata (con liste collegate) è presente una certa perdita di spazio dovuto alla frammentazione interna.
- C. L'allocazione contigua è la soluzione che richiede meno memoria RAM ed il minor numero di accessi al disco per determinare il blocco in cui è memorizzato un arbitrario contenuto all'interno di un file.
- D. Usando una FAT per tenere traccia dei blocchi dei file non è necessario mantenere una ulteriore bitmap per tenere traccia dei blocchi liberi.
- E. Nell'allocazione che fa uso della tabella di allocazione dei file (FAT) la capacità del singolo blocco su disco può essere solo parzialmente sfruttata per memorizzare i contenuti del file, dovendo memorizzare il numero del blocco successivo.

- 3. Consideriamo un sistema che fa uso di memoria virtuale con le seguenti caratteristiche: uno spazio di indirizzamento virtuale da 1 Gb, un numero di pagina virtuale a 22 bit e un indirizzo fisico a 20 bit. Determinare esattamente quanti frame fisici ci sono in memoria.**

4096 Frame da 2<sup>8</sup> B

- 4. Supponiamo di avere 3 processi che condividono una variabile x e che i loro pseudo-codici siano i seguenti:**

P1:

wait(S)

x=x-2

signal(T)

wait(S)

x=x-1

signal(T)

P2:

wait(R)

x=x+2

signal(T)

wait(R)

P3:

wait(T)

if (x<0) signal(R)

wait(T)

print(x)

**Determinare l'output del processo P3 assumendo che il valore iniziale di x è 1 e che i 3 semafori abbiano i seguenti valori iniziali: S=1, R=0, T=0.**

X = 1

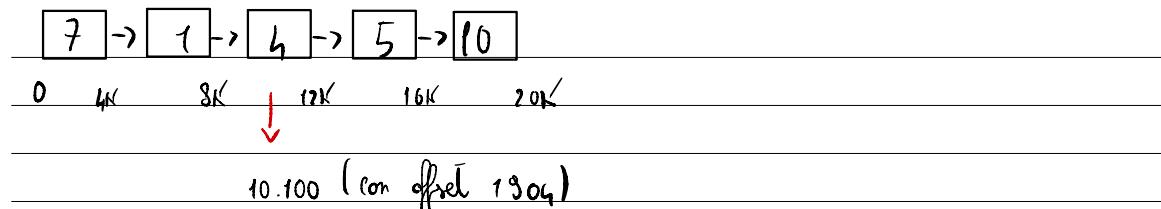
5. Supponiamo di avere un file-system che utilizza per tenere traccia dei file in esso memorizzati la seguente FAT e che prevede le seguenti voci all'interno della cartella radice:

| FAT    | cartella radice |
|--------|-----------------|
| indice |                 |
| 1      | 4               |
| 2      | 3               |
| 3      | 15              |
| 4      | 5               |
| 5      | 10              |
| 6      | 12              |
| 7      | 1               |
| 8      | 2               |
| 9      | 3               |
| 10     | -1              |
| ...    | ...             |

| nome      | primo blocco |
|-----------|--------------|
| ...       | ...          |
| pippo.txt | 7            |
| ...       | ...          |

Indicare esattamente in quale blocco del disco (indicare il numero di blocco) è localizzabile l'offset 10100 all'interno del file pippo.txt. Indicare qual'è la dimensione minima presunta in byte dello stesso file. In tale calcolo tenere conto del fatto che un blocco del file-system è grande 4 kB.

Nota: gli offset sono espressi in byte e partono da 0.



6. [duplicato] Consideriamo un file-system UNIX basato su i-node: l'i-node di un file contiene, oltre ad una serie di meta-dati, un certo numero di voci che servono ad individuare i blocchi del disco su cui è memorizzato il contenuto del file stesso. In un i-node standard ci sono 13 di queste voci: le ultime 3 sono usate per indicare, rispettivamente, un blocco indiretto singolo, un blocco indiretto doppio ed, per ultimo, un blocco indiretto triplo. Prendiamo come esempio il seguente i-node ed il contenuto di alcuni blocchi sul disco (di alcuni blocchi dati sono indicate solo le word preliminari e finali):

| i-node 54             | blocco 112 | blocco 444 | blocco 333 | blocco 233 | blocco 322 |
|-----------------------|------------|------------|------------|------------|------------|
| meta-dati<br>del file |            |            |            |            |            |
| 321                   | 16         | 200        | 233        | 821        | 323        |
| 322                   | 544        | 288        | 322        | 822        | 212        |
| 239                   | 20         | 201        | 444        | 915        | 999        |
| 234                   | 555        | 280        | 530        | 50         | 0          |
| 235                   | 922        | 399        | 742        | 51         | 843        |
| 236                   | 942        | 400        | 221        | 53         | 212        |
| 14                    | ...        | ...        | ...        | ...        | ...        |
| 21                    | ...        | ...        | ...        | ...        | ...        |
| 233                   | ...        | ...        | ...        | ...        | ...        |
| 12                    | 132        | 899        | -1         | 881        | 233        |
| 112                   | 134        | 900        | -1         | 882        | 0          |
| 333                   |            |            |            |            |            |
| -1                    |            |            |            |            |            |

Tenendo conto del fatto che i blocchi usati dal file-system sono da 4 kB e che i numeri di blocco sono a 32 bit: individuare in quali blocchi del disco (indicare il numero di blocco) risiedono i byte di offset XXX, YYY, ZZZ del contenuto del file a cui si riferisce l'i-node.

Nota: gli offset sono espressi in byte e partono da 0.

7. Supponiamo di avere un disco con 200 tracce (numerate da 0 a 199) la cui velocità di seek è di 1 traccia per ms. All'istante  $t=0$  il sistema operativo sta servendo una richiesta sulla traccia 100 e in coda ci sono già le seguenti richieste per le tracce (50, 115, 180). Successivamente arrivano altre richieste all'istante  $t=70$  per la traccia 150 e all'istante  $t=130$  per la traccia 90. Si calcoli il tempo di ricerca complessivo (in ms) per servire tutte le richieste secondo la politica LOOK, iniziando in ordine ascendente (dalla traccia 0 verso la traccia 199) e trascurando la latenza rotazionale e il tempo di trasferimento. Indicare anche la sequenza di scheduling considerata.

100 ← 115 ← 180 ← 150 ← 50 ← 90

---

3. Consideriamo un sistema che fa uso di memoria virtuale con le seguenti caratteristiche: uno spazio di indirizzamento virtuale da 1 Gb, un numero di pagina virtuale a 22 bit e un indirizzo fisico a 20 bit. Determinare esattamente quanti frame fisici ci sono in memoria.

$$VAS = 1GB = 2^{30} B$$

$$\# PV = 22 \text{ b}$$

$$IF = 20 \text{ b}$$

# Frame Fisici ?

$$N = 30 - 22 = 8 \text{ b}$$

$$U - N = 20 - 8 = 12 \text{ b}$$

$$\Rightarrow 2^{12} = 4096 \text{ frame fisici} \\ (\text{da } 2^8 \text{ B})$$

7. Supponiamo di avere un disco con 200 tracce (numerate da 0 a 199) la cui velocità di seek è di 1 traccia per ms. All'istante  $t=0$  il sistema operativo sta servendo una richiesta sulla traccia 100 e in coda ci sono già le seguenti richieste per le tracce (50, 115, 180). Successivamente arrivano altre richieste all'istante  $t=70$  per la traccia 150 e all'istante  $t=130$  per la traccia 90. Si calcoli il tempo di ricerca complessivo (in ms) per servire tutte le richieste secondo la politica LOOK, iniziando in ordine ascendente (dalla traccia 0 verso la traccia 199) e trascurando la latenza rotazionale e il tempo di trasferimento. Indicare anche la sequenza di scheduling considerata.

Sequenza:  $100 \leftarrow 115 \leftarrow 180 \leftarrow 150 \leftarrow 50 \leftarrow 90$

