

Sistemi Operativi

Esercizi Gestione Memoria

Docente: Claudio E. Palazzi
cpalazzi@math.unipd.it

Esercizio 1

Dato un sistema di *swapping* e una memoria con zone disponibili di ampiezza: 8, 4, 14, 18, 17, 9, 12, 15 KB, in questo ordine, indicare quale area venga prescelta dalla politica ***Next Fit*** a fronte della richiesta di caricamento di un segmento di ampiezza 3 KB dopo aver caricato un segmento ampio 12 KB:

A: 4 KB

B: 18 KB

C: 14 KB

D: 9 KB.

Esercizio 1 - soluzione

Dato un sistema di *swapping* e una memoria con zone disponibili di ampiezza: 8, 4, 14, 18, 17, 9, 12, 15 KB, in questo ordine, indicare quale area venga prescelta dalla politica ***Next Fit*** a fronte della richiesta di caricamento di un segmento di ampiezza 3 KB dopo aver caricato un segmento ampio 12 KB:

A: 4 KB

 B: 18 KB

C: 14 KB

D: 9 KB

E con ***First fit? Best fit? Worst fit?***

Esercizio 2

Sia data memoria dotata di 4 *page frame*, inizialmente libere, e 8 pagine di memoria virtuale. Utilizzando la politica FIFO per il rimpiazzo delle pagine, indicare quanti *page fault* si verifichino a fronte della stringa di riferimenti: 0 1 7 2 3 2 7 1 0 3:

A: 4

B: 3

C: 6

D: 2

Esercizio 2 - soluzione

Sia data memoria dotata di 4 *page frame*, inizialmente libere, e 8 pagine di memoria virtuale. Utilizzando la politica FIFO per il rimpiazzo delle pagine, indicare quanti *page fault* si verifichino a fronte della stringa di riferimenti: 0 1 7 2 3 2 7 1 0 3:

- A: 4
- B: 3
-  C: 6
- D: 2

Riferimenti	0	1	7	2	3	2	7	1	0	3
RAM	0	1	7	2	3	3	3	3	0	0
		0	1	7	2	2	2	2	3	3
			0	1	7	7	7	7	2	2
				0	1	1	1	1	7	7
	P	P	P	P	P				P	

6 PF

Esercizio 2 - soluzione

Sia data memoria dotata di 4 *page frame*, inizialmente libere, e 8 pagine di memoria virtuale. Utilizzando la politica FIFO per il rimpiazzo delle pagine, indicare quanti *page fault* si verifichino a fronte della stringa di riferimenti: 0 1 7 2 3 2 7 1 0 3:

A: 4

B: 3

 C: 6

D: 2

Posso decidere anche per altre politiche di rimpiazzo (NRU, *Second Chance*, LRU, ecc.)? Oppure ho bisogno di altre informazioni (in caso affermativo, quali?) ?

Esercizio 3

Si consideri la seguente serie di riferimenti a pagine di memoria:

1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6.

Si considerino le seguenti politiche di rimpiazzo:

- FIFO
- LRU
- Optimal

Quanti *page fault* avvengono considerando un numero di *page frame* della RAM di 1, 2, 3, 4, 5, 6, 7 ?

Esercizio 3 - FIFO

Riferimenti	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
Meno vecchio	1	2	3	4	4	1	5	6	2	1	1	3	7	6	6	2	1	1	3	6
RAM		1	2	3	3	4	1	5	6	2	2	1	3	7	7	6	2	2	1	3
più vecchio			1	2	2	3	4	1	5	6	6	2	1	3	3	7	6	6	2	1
	P	P	P	P		P	P	P	P	P		P	P	P		P	P		P	P

16 PF

Riferimenti	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
RAM	1	2	3	4	4	4	5	6	2	1	1	3	7	6	6	2	1	1	3	3
		1	2	3	3	3	4	5	6	2	2	1	3	7	7	6	2	2	1	1
			1	2	2	2	3	4	5	6	6	2	1	3	3	7	6	6	2	2
				1	1	1	2	3	4	5	5	6	2	1	1	3	7	7	6	6
	P	P	P	P		P	P	P	P		P	P	P		P	P		P		

14 PF

Esercizio 3 - LRU

Riferimenti	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
Rif. + recentem.	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
RAM		1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3
Rif. - recentem.			1	2	3	4	2	1	5	6	6	1	1	3	7	6	3	3	1	2
	P	P	P	P		P	P	P	P	P		P	P	P		P	P			P

15 PF

Riferimenti	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
RAM	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
		1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3
			1	2	3	4	2	1	5	6	6	1	2	3	7	6	3	3	1	2
				1	1	3	4	2	1	5	5	6	1	2	2	7	6	6	6	1
	P	P	P	P		P	P				P	P	P			P				

10 PF

Esercizio 3 - OPTIMAL

Riferimenti	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6	
RAM	1	1	1	1	1	1	1	1	1	1	1	3	3	3	3	3	3	3	3	6	
		2	2	2	2	2	2	2	2	2	2	2	7	7	7	2	2	2	2	2	
			3	4	4	4	5	6	6	6	6	6	6	6	6	6	6	1	1	1	1
	P	P	P	P			P	P			P	P			P	P				P	

Ultimo 6 riferito può essere inserito in qualsiasi posizione

11 PF

Riferimenti	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
RAM	1	1	1	1	1	1	1	1	1	1	1	1	7	7	7	7	1	1	1	1
		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
			3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
				4	4	4	5	6	6	6	6	6	6	6	6	6	6	6	6	6
	P	P	P	P			P	P			P					P				

8 PF

Esercizio 3 - Soluzioni

<i># page frame</i>	FIFO	LRU	<i>Optimal</i>
1	20	20	20
2	18	18	15
3	16	15	11
4	14	10	8
5	10	8	7
6	10	7	7
7	7	7	7

Esercizio 4

Sia dato un sistema di gestione della memoria principale basato su segmentazione con indirizzi logici e fisici espressi su 16 *bit*. Si consideri la tabella dei segmenti riportata di seguito, ove il prefisso 0x denota l'uso di notazione Esadecimale:

Segmento	Base	Limite
0x0	0x0219	0x600
0x1	0x2300	0x014
0x2	0x0090	0x100
0x3	0x1327	0x580
0x4	0x1952	0x096
...
0xF

Si mostri graficamente la mappa di memoria corrispondente, indicando anche il suo grado percentuale di occupazione.

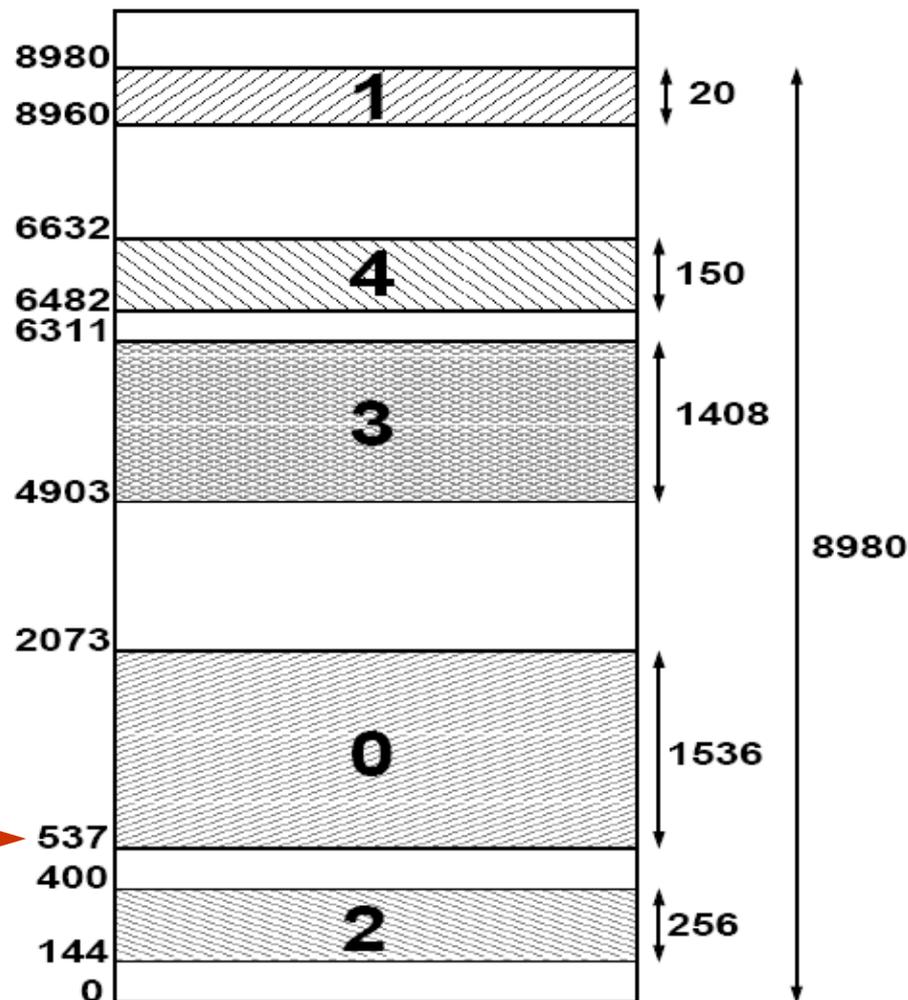
Esercizio 4 - soluzione

Con un po' di calcoli:

$$\begin{aligned} 0 \times 219 &= \\ 2 \times 256 + 1 \times 16 + 9 \times 1 &= \\ \mathbf{537} \end{aligned}$$

Eccetera ...

L'area di memoria occupata va dall'indirizzo 0 all'indirizzo 8980.
L'ampiezza complessiva dei segmenti in tale zona misura 3370 B con grado di occupazione: **37, 53%**.



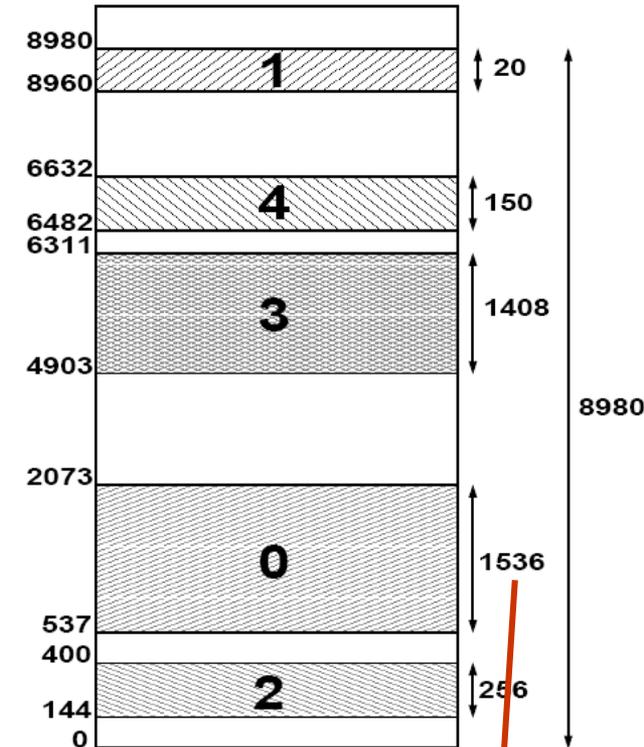
Esercizio 4.1

Si fornisca l'indirizzo fisico corrispondente ai seguenti indirizzi virtuali espressi in notazione decimale:

-
- a. $\langle 0, 1984 \rangle$
 - b. $\langle 1, 18 \rangle$
 - c. $\langle 2, 250 \rangle$
 - d. $\langle 3, 1400 \rangle$
 - e. $\langle 4, 112 \rangle$
-

Esercizio 4.1 - soluzione

Ricordando che ...



Allora:

indirizzo virtuale	indirizzo fisico
< 0, 1984 >	Errore: indirizzo illegale.
< 1, 18 >	$8960 + 0018 = 8978$
< 2, 250 >	$0144 + 0250 = 0394$
< 3, 1400 >	$4903 + 1400 = 6303$
< 4, 112 >	$6482 + 0112 = 6594$

1984 > 1536

Esercizio 4.2

Indicare il modo nel quale i segmenti di indice 0–4 possano essere mappati su disco, assumendo blocchi di ampiezza 1 KB, calcolando anche la quantità percentuale di memoria *sprecata* di conseguenza.

Esercizio 4.2 - soluzione

Blocchi di ampiezza 1 KB = 1024 B, quindi:

Segmento	Ampiezza	# Blocchi occupati
0	1536 B	2
1	20 B	1
2	256 B	1
3	1408 B	2
4	150 B	1
Totale	3370 B	7

$$\text{Totale B} \times \text{Blocchi} = 1024 \text{ B} \times 7 = 7168 \text{ B}$$

$$\text{Totale spreco} = (7168 - 3370) \text{ B} = 3798 \text{ B}$$

$$\text{Spreco di memoria: } 3798 \text{ B} / 7168 \text{ B} = \mathbf{53\%}$$

Esercizio 5

Si consideri la matrice (o *array* bidimensionale):

```
int A[][] = new int[100][100];
```

Si assuma che la posizione **A[0][0]** di tale matrice sia posta alla locazione **200** di una memoria paginata con **3** pagine di dimensione **200 B**.

Si assuma che un valore `int` occupi **1 B**.

Si assuma che le pagine siano inizialmente tutte vuote e che il processo (il cui codice occupa esattamente **200 B**) abbia la seguente esecuzione:

```
for (int i = 0; i < 100; i++) {  
    for (int j = 0; j < 100; j++) {  
        A[i][j] = 0; } } }
```

Quanti *page fault* saranno generati usando LRU?

Esercizio 5 – soluzione (1/2)

La matrice $A[i][j]$ è scritta linearmente in memoria:

$A[0][0], A[0][1], A[0][2], \dots, A[0][99], A[1][0],$
 $A[1][1], \dots, A[99][99]$

Il processo azzerava le celle della matrice proprio nel loro ordine di memorizzazione.

Quindi inizialmente verranno caricati nelle pagine

Pagina 0: Il programma (che occupa esattamente 200 B)

Pagina 1: Celle da $A[0][0]$ a $A[1][99]$ incluse

Pagina 2: Celle da $A[2][0]$ a $A[3][99]$ incluse

Esercizio 5 – soluzione (2/2)

A ogni iterazione del programma la memoria viene acceduta per leggere l'istruzione successiva

→ la pagina relativa al processo verrà continuamente acceduta

I primi 200 azzeramenti faranno accesso a Pagina 1, i successivi 200 a Pagina 2; l'ulteriore azzeramento (cella $A[4][0]$) causerà un *page fault*; la pagina usata meno recentemente è Pagina 1 che verrà sostituita con le celle da $A[4][0]$ a $A[5][99]$ incluse. Arrivati all'azzeramento di $A[6][0]$ sarà Pagina 2 ad essere stata usata meno di recente

E così via, causando in tutto 1 *page fault* iniziale per caricare il processo in Pagina 0 e 50 *page fault* di Pagina 1 e 2 (25 ciascuno) per caricare le porzioni di matrice.

TOTALE = 51 *page fault*

Esercizio 6 (1/2)

Si consideri un sistema dotato di memoria virtuale, con memoria fisica divisa in 8 *page frame* condivisa da 4 processi contemporaneamente attivi: A, B, C e D. Si supponga che all'istante 100 lo stato della memoria sia il seguente (con il campo riferita avente SI = 1 e NO = 0):

Processo	pag. logica	pag. fisica	istante caricam.	R
A	0	7	50	1
A	1	6	37	0
B	5	5	30	1
B	3	4	97	0
C	2	0	15	1
C	5	2	70	0
C	9	1	92	0
D	8	3	27	0

Esercizio 6 (1/2)

Si supponga che il sistema utilizzi un algoritmo di sostituzione **second chance (globale)**, e che la lista delle pagine accedute all'istante 100 sia:

C2(15,1) - D8(27,0) - B5(30,1) - A1(37,0) - A0(50,1) - C5(70,0) - C9(92,0) - B3(97,0)

dove C2(15, 1) sia il primo elemento della lista; in tale elemento, C2 indica che si tratta della pagina logica 2 del processo C, mentre (15, 1) indica che tale pagina è stata caricata in memoria all'istante 15 e che il suo bit riferita è uguale a 1.

Si considerino i due seguenti casi, eseguiti in sequenza:

Caso a) all'istante 101, C riferisce la pagina logica 5

Caso b) all'istante 105, A riferisce la pagina logica 9

Assumendo che non vi siano altre operazioni che modifichino i bit riferita, a parte quelle sopra elencate, si scriva la lista delle pagine accedute aggiornata dopo aver eseguito, in successione, i punti a) e b).

Esercizio 6 – Soluzione

Caso a) La pagina C5 è già in memoria (nella pagina fisica 2) e quindi viene marcata come riferita

La lista risultante è dunque:

C2(15,1) - D8(27,0) - B5(30,1) - A1(37,0) - A0(50,1) - C5(70,1) - C9(92,0) - B3(97,0)

Caso b) La pagina logica A9 non si trova in memoria, l'algoritmo *second chance* esamina C2: siccome è stata riferita, viene inserita in coda alla lista ed il campo riferita viene posto uguale a 0; dopodichè, si esamina D8, che non risulta riferita e viene dunque eliminata. La lista risultante è dunque:

B5(30,1) - A1(37,0) - A0(50,1) - C5(70,1) - C9(92,0) - B3(97,0) - C2(15,0) - A9(105,1)

Esercizio 6 – Soluzione bis

E se la politica di rimpiazzo fosse stata **locale**?

Allora è come se ci fossero liste separate e in particolare quella di C risulta $C2(15,1) - C5(70,0) - C9(92,0)$

Quindi per il **Caso a)**, siccome C5 è già in memoria (nella pagina fisica 2), essa viene marcata come riferita e la lista risultante è dunque: $C2(15,1) - C5(70,1) - C9(92,0)$

Invece, per A, la lista corrispondente sarebbe: $A1(37,0) - A0(50,1)$

Quindi per il **Caso b)**, siccome la pagina logica A9 non si trova in memoria, l'algoritmo *second chance* esamina A1 che non risulta riferita e viene dunque rimpiazzata.

La lista risultante è dunque: $A0(50,1) - A9(105,1)$

Per completezza, se la lista (per ipotesi) fosse stata $A1(37,1) - A0(50,0)$

Allora A1 sarebbe stato portato in fondo con bit riferito azzerato e poi A0 sarebbe stato rimpiazzato, inserendo comunque A9 in fondo, con risultato: $A1(37,0) - A9(105,1)$