SISTEMI OPERATIVI, CORSI A e B Prova del 16/12/2009 CORSO |A| |B|

COGNOME E NOME ..... MATRICOLA .....;..... AULA ___ FILA ___ POSTO ___

ESERCIZIO 1 (4 punti)

Un sistema con 20 MB di memoria principale gestisce la memoria con rilocazione dinamica, caricamento in partizioni variabili e allocazione della memoria con politica best fit. Una partizione con origine 0 e dimensione 1MB è riservata al sistema operativo, mentre i restanti 19 MB sono disponibili per i processi utente.

All'istante di tempo t sono caricati in memoria i seguenti processi:

- il processo A, che risiede in memoria dal tempo t–300 ed occupa una partizione con origine 1 e lunghezza 3 MB;
- il processo B, che risiede in memoria dal tempo t–200 ed occupa una partizione con origine 6 e lunghezza 5 MB;
- il processo C, che risiede in memoria dal tempo t–100 ed occupa una partizione con origine 15 e lunghezza 2 MB.

Successivamente sono generati i seguenti processi:

- al tempo t+ 100 il processo D che richiede una partizione di lunghezza 3 MB;
- al tempo t+ 200 il processo E che richiede una partizione di lunghezza 5 MB;
- al tempo t+ 300 il processo F che richiede una partizione di lunghezza 7 MB;
- al tempo t+ 400 il processo G che richiede una partizione di lunghezza 1 MB.

Il codice dei processi è rilocabile e i processi vengono caricati in memoria in ordine FIFO. Tutti i tempi sono espressi in *msec* e tutte le lunghezze delle partizioni sono espresse in *Mbyte*.

Quando viene generato un processo il gestore della memoria applica la politica *Best Fit* per selezionare e, se esiste, per assegnare una partizione di lunghezza sufficiente.

Se non esiste una singola partizione libera di lunghezza sufficiente, ma la somma delle lunghezze dei frammenti è maggiore o uguale all'esigenza del processo, allora il gestore della memoria procede al compattamento della memoria (verso il basso) e quindi carica il processo. Altrimenti, se la lunghezza complessiva dei frammenti non è sufficiente, il gestore della memoria procede allo swap-out di uno più processi, fino a liberare una partizione di lunghezza sufficiente per il caricamento. Lo swap out procede scaricando uno ad uno i processi in ordine decrescente di lunghezza della partizione occupata ed eseguendo il compattamento dopo lo scaricamento di ogni singolo processo. A parità di lunghezza della partizione occupata è scaricato per primo il processo che risiede in memoria da più tempo.

Utilizzando il grafico sotto riportato, mostrare come evolve l'occupazione della memoria nei tempi t+100, t+200, t+300 e t+400 e come evolve la coda dei processi in attesa di caricamento.

Si suppone che nessuno dei processi termini in questo intervallo di tempo e che le operazioni di compattamento e swap-out si completino un tempo trascurabile.

SOLUZIONE

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
t		A	A	A			B	B	B	B	B					C	C			
t+100		A	A	A			B	B	B	B	B					C	C	D	D	D
t+200		A	A	A	B	B	B	B	B	C	C	D	D	D	E	E	E	E	E	
t+300		A	A	A	C	C	D	D	D	F	F	F	F	F	F	F				
t+400		A	A	A	C	C	D	D	D	F	F	F	F	F	F	F	G			

Operazioni eseguite:

t+100: Processi swapped --	Eseguito compattamento NO	Processo caricato D
t+200: Processi swapped --	Eseguito compattamento SI	Processo caricato E
t+300: Processi swapped B, E	Eseguito compattamento SI	Processo caricato F
t+400: Processi swapped --	Eseguito compattamento NO	Processo caricato G

SISTEMI OPERATIVI, CORSI A e B Prova del 16/12/2009 CORSO |A| |B|

ESERCIZIO 2 (4 punti)

In un sistema che gestisce la memoria con paginazione a domanda, le pagine logiche e i blocchi fisici hanno una lunghezza di 2^6 byte e gli indirizzi logici hanno una lunghezza di 14 bit.

Per la gestione della memoria si utilizzano tabelle delle pagine a 2 livelli. Sia la tabella di primo livello che quelle di secondo livello contengono 2^4 elementi di 4 byte: pertanto ogni tabella occupa una pagina e la componente dell'indirizzo logico che individua le pagine è suddivisa in due parti di 4 bit ciascuna, denominate indice di primo livello (che indicizza la tabella di primo livello) e indice di secondo livello (che indicizza la tabella di secondo livello).

La tabella di primo livello è caricata permanentemente in memoria principale, in un blocco noto al sistema operativo, mentre le tabelle di secondo livello sono caricate a domanda.

L'elemento della tabella di primo livello di indice *i* è il descrittore della tabella di secondo livello di indice *i*, e contiene l'indicatore *P* di presenza della tabella ed eventualmente il blocco di memoria fisica nel quale la tabella medesima è caricata.

L'elemento della tabella di secondo livello di indice *j* è il descrittore della pagina *j* e contiene l'indicatore *P* di presenza della pagina ed eventualmente il blocco di memoria fisica nel quale la pagina è caricata.

Al tempo t è in esecuzione il processo P e i contenuti parziali della sua tabella delle pagine di primo livello e delle tabelle di secondo livello caricate in memoria principale sono mostrati in figura. Nella memoria fisica sono disponibili alcuni blocchi, che al verificarsi di *page faults* sono utilizzabili per caricare tabelle di secondo livello o pagine del processo. Questi blocchi sono ordinati nella coda <PrimoBlocco> → 90 → 91 → 92 → 93 → 94 → 95…….. e, in caso di *page fault*, sono utilizzati in questo ordine.

Indice 1° Liv	Blocco	P
0000	--	0
0001	--	0
0010	31	1
0011	--	0
0100	35	1
0101	--	0
0110	34	1
0111	--	0
…….	…	…

Tabella 1° Livello

Indice 2° Liv	Blocco	P
0000	--	0
0001	40	1
0010	--	0
0011	--	0
0100	--	0
0101	--	0
0110	41	1
0111	--	0
…….	…	…

Tabella 2° Livello Indice 0010

Indice 2° Liv	Blocco	P
0000	--	0
0001	--	0
0010	50	1
0011	--	0
0100	--	0
0101	--	0
0110	--	0
0111	--	0
…….	…	…

Tabella 2° Livello Indice 0100

Indice 2° Liv	Blocco	P
0000	63	1
0001	--	0
0010	64	1
0011	--	0
0100	--	0
0101	--	0
0110	--	0
0111	--	0
…….	…	…

Tabella 2° Livello Indice 0110

A partire dal tempo t il processo P accede alla memoria in sequenza con i seguenti indirizzi binari, nei quali è omessa la componente di 6 bit riservata all'offset nella pagina:

1. indirizzo 0000 0010
2. indirizzo 0010 0000
3. indirizzo 0111 0110
4. indirizzo 0111 0100
5. indirizzo 0110 0010
6. indirizzo 0000 0010

Per ciascun accesso alla memoria, si chiede: se l'accesso alla tabella di secondo livello determina page fault

- il blocco fisico che contiene (eventualmente dopo il caricamento in memoria) la tabella di secondo livello
- se l'accesso alla pagina riferita determina page fault
- blocco fisico che contiene (eventualmente dopo il caricamento in memoria) la pagina riferita.

Ipotesi: Quando una tabella di secondo livello è caricata in memoria, in ogni suo descrittore il bit di presenza vale 0.
Nell'intervallo di tempo considerato (accessi 1 .. 6) il gestore della memoria non scarica nessuna pagina.

SOLUZIONE

1) Indirizzo 0000 0010

l'accesso alla tabella di 2° livello determina page fault?	SI
blocco fisico che contiene la tabella di secondo livello	90
l'accesso alla pagina riferita determina page fault?	SI
blocco fisico che contiene la pagina riferita	91

2) Indirizzo 0010 0000

l'accesso alla tabella di 2° livello determina page fault?	NO
blocco fisico che contiene la tabella di secondo livello	31
l'accesso alla pagina riferita determina page fault?	SI
blocco fisico che contiene la pagina riferita	92

3) Indirizzo 0111 0110

l'accesso alla tabella di 2° livello determina page fault?	SI
blocco fisico che contiene la tabella di secondo livello	93
l'accesso alla pagina riferita determina page fault?	SI
blocco fisico che contiene la pagina riferita	94

4) Indirizzo 0111 0100

l'accesso alla tabella di 2° livello determina page fault?	NO (*)
blocco fisico che contiene la tabella di secondo livello	93
l'accesso alla pagina riferita determina page fault?	SI
blocco fisico che contiene la pagina riferita	95

(*) Tabella caricata al passo 3) nel blocco 93

5) Indirizzo 0110 0010

l'accesso alla tabella di 2° livello determina page fault?	NO
blocco fisico che contiene la tabella di secondo livello	34
l'accesso alla pagina riferita determina page fault?	NO
blocco fisico che contiene la pagina riferita	64

6) Indirizzo 0000 0010

l'accesso alla tabella di 2° livello determina page fault?	NO (**)
blocco fisico che contiene la tabella di secondo livello	90
l'accesso alla pagina riferita determina page fault?	NO (***)
blocco fisico che contiene la pagina riferita	91

(**) Tabella caricata al passo 1) nel blocco 90
(***) pagina caricata al passo 1) nel blocco 91

SISTEMI OPERATIVI, CORSI A e B Prova del 16/12/2009 CORSO |A| |B|

ESERCIZIO 3 (4 punti)

In un file system UNIX i blocchi del disco hanno ampiezza di 512 byte e gli i-node contengono 10 indirizzi diretti e 3 indirizzi indiretti. Tutti gli indirizzi hanno una lunghezza di 32 bit.
Dati i file A, B e C di dimensione rispettivamente 65.000 byte, 890.000 byte e 1.000.000 byte, calcolare per ognuno di questi file il numero di blocchi indice di primo, secondo e terzo livello necessari per rappresentarlo.
Si chiede anche di determinare preliminarmente il numero di puntatori che possono essere contenuti in ogni blocco indice, il numero di blocchi indirizzabili con indirizzamento indiretto semplice, con indirizzamento indiretto doppio e con indirizzamento indiretto triplo.

SOLUZIONE

1. Il numero di puntatori contenuti in un blocco indice è 512 / 4 = 128. Pertanto:
   - il numero di blocchi indirizzabili con indirizzamento indiretto semplice è: 2^7= 128
   - il numero di blocchi indirizzabili con indirizzamento indiretto doppio è: 2^14= 16.384
   - il numero di blocchi indirizzabili con indirizzamento indiretto triplo è: 2^21= 2.097.152

   Si nota che, data la dimensione dei file A, B e C, in nessun caso si deve ricorrere all'indirizzamento indiretto triplo.
2. Il file A occupa $\lceil 65.000$ **div** $512 \rceil = 127$ blocchi dati,, dei quali:
   - 10 sono indirizzati dai puntatori diretti (indici 0 .. 9 nello i-node),
   - 117 sono indirizzati attraverso un unico blocco indice (di primo livello), a sua volta indirizzato dal puntatore indiretto semplice (indice 10 nello i-node).

   → Pertanto per la rappresentazione del file è necessario 1 blocco indice.
3. Il file B occupa $\lceil 890.000$ **div** $512 \rceil = 1739$ blocchi, dei quali:
   - 10 sono indirizzati dai puntatori diretti (indici 0 .. 9 nello i-node),
   - 1729 sono indirizzati attraverso $\lceil 1729$ **div** $128 \rceil = 14$ blocchi indice (di primo livello), dei quali:

   il primo, che indirizza 128 blocchi dati, è a sua volta indirizzato dal puntatore indiretto semplice (indice 10 nello i-node).
   - i rimanenti 13 blocchi indice (di primo livello) sono indirizzati dai puntatori di indice 0 .. 12 di un unico blocco indice (di secondo livello), a sua volta indirizzato dal puntatore indiretto doppio (indice 11 nello i-node).

   → Pertanto per la rappresentazione del file sono necessari in totale 15 blocchi indice (14 di primo livello e 1 di secondo livello)
4. Il file C occupa $\lceil 1.000.000$ **div** $512 \rceil = 1954$ blocchi logici, dei quali:
   - 10 sono indirizzati dai puntatori diretti (indici 0 .. 9 nello i-node),
   - 1944 sono indirizzati attraverso $\lceil 1944$ **div** $128 \rceil = 16$ blocchi indice (di primo livello), dei quali:

   il primo, che indirizza 128 blocchi dati, è a sua volta indirizzato dal puntatore indiretto semplice (indice 10 nello i-node).
   - i rimanenti 15 blocchi indice sono indirizzati dai puntatori di indice 0 .. 14 di un unico blocco indice (di secondo livello), a sua volta indirizzato dal puntatore indiretto doppio (indice 11 nello i-node).

   → Pertanto per la rappresentazione del file sono necessari in totale 17 blocchi indice (16 di primo livello e 1 di secondo livello).

SISTEMI OPERATIVI, CORSI A e B Prova del 16/12/2009 CORSO |A| |B|

ESERCIZIO 4 (4 punti)

Un disco RAID di livello 0 è composto da 3 dischi fisici indipendenti, individuati dagli indici 0, 1 e 2. Ogni disco fisico ha 120 cilindri, 4 facce e 30 settori per traccia, con tempo di seek proporzionale al numero di cilindri attraversati e uguale a 0,5ms per ogni cilindro. Il periodo di rotazione è di 3 msec: conseguentemente ogni settorte viene percorso in 0,1 msec.

I blocchi, che corrispondono ai settori, contengono 1024 caratteri e le strip coincidono con i blocchi. Il blocco *b* del disco virtuale V è mappato nel blocco *b'= b* ***div*** *3* del disco fisico di indice *f= b* ***mod*** *3*. Per l'ordinamento dei comandi pendenti, ciascuno dei dischi fisici adotta la politica SCAN.

Al tempo *t* si ha la seguente situazione:

- il disco di indice *f= 0* è in fase di discesa, ha iniziato l'esecuzione di un'operazione sul cilindro 10 che terminerà al tempo t+2 e non ha altri comandi pendenti;
- il disco di indice *f= 1* è in fase di discesa, ha iniziato l'esecuzione di un'operazione sul cilindro 50 che terminerà al tempo t+10 e non ha altri comandi pendenti;
- il disco D2 è in fase di salita, ha iniziato l'esecuzione di un'operazione sul cilindro 40 che terminerà al tempo t+5 e non ha altri comandi pendenti;

Successivamente:

- Al tempo t+1 il disco virtuale V riceve un comando per la lettura di 4 blocchi consecutivi a partire da quello di indice virtuale 270;
- Al tempo t+4 il disco virtuale V riceve un comando per la lettura di 5 blocchi consecutivi a partire da quello di indice virtuale 30.000;
- dopo il tempo t+4 e fino al tempo 100 il disco virtuale non riceve altri comandi.

Si chiede:

1. La capacità in blocchi di ciascuno dei dischi fisici;
2. La capacità in blocchi del disco virtuale V;
3. Come sono mappati sui dischi fisici i comandi di lettura che il disco virtuale V riceve ai tempi t+ 1 e t+ 4. Per ogni disco fisico, individuare i blocchi da leggere sotto forma di terna *(clilindro, faccia, settore)*;
4. Il tempo di inizio e fine di ciascuna operazione eseguita da ogni disco fisico. Per semplicità, dopo un'operazione di seek si assume sempre un ritardo rotazionale di 3 msec, qualunque sia il settore da raggiungere.

SOLUZIONE

1. Capacità in blocchi di ciascun disco fisico: 120* 4* 30= 14.400 blocchi;
2. Capacità in blocchi del disco virtuale: 14.400* 3= 43.200;
3. Dalle formule *f= b* ***mod*** *3; b'= b* ***div*** *3; cilindro= b'* div *(4*30), faccia=(b'* mod *(4*30))* div *30; settore= (b'* mod *(4*30))* mod *30,* il mappaggio sui dischi fisici dei blocchi da leggere è il seguente:
   - disco fisico di indice 0:
     - al tempo t+1, blocco b'= 90→ (cil= 0, faccia= 3, sett = 0) e blocco b'= 91→ (cil = 0, faccia= 3, sett = 1) *;*
     - al tempo t+ 4, blocco b'= 10000→ (cil = 83, faccia=1, sett =10) e blocco b'= 10001→ (cil= 83, faccia=1, sett = 11)*;*
   - disco fisico di indice 1:
     - al tempo t+1 blocco b'= 90→ (cil = 0, faccia= 3, sett = 0);
     - al tempo t+ 4, blocco b'= 10000→ (cil=83, faccia=1, sett= 10) e blocco b'= 10001→ (cil= 83, faccia= 1, sett = 11) *;*
   - disco fisico di indice 2:
     - al tempo t+1 blocco b'= 90→ (cil = 0, faccia= 3, sett = 0);
     - al tempo t+ 4, blocco b'= 10000→ (cil = 83, faccia= 1, sett = 10)

(continua alla pagina successiva)

4. Tempo di inizio e fine delle operazioni eseguite dai dischi fisici:

Disco di indice 0: al tempo t+2 sono pendenti comandi per blocchi (0, 3, 0) e (0, 3, 1); al tempo t+4 arrivano comandi per blocchi (83, 1, 10) e (83, 1, 11)

1) op. su cilindro:	0	settore:	0	faccia :	3				
inizio:	2	seek:	10*0,5	rotazione:	3	percorrenza:	0,1	fine:	10,1
2) op. su cilindro:	0	settore	1	faccia :	3				
inizio:	8,1	seek:	0	rotazione:	0	percorrenza:	0,1	fine:	10,2
3) op. su cilindro:	83	settore:	10	faccia :	1				
inizio:	5,2	seek:	(83-0)*0,5	rotazione:	3	percorrenza:	0,1	fine:	54,8
4) op. su cilindro:	83	settore:	11	faccia :	1				
inizio:	49,8	seek:	0	rotazione:	0	percorrenza:	0,1	fine:	54,9

Disco di indice 1: al tempo t+10 sono pendenti comandi per blocchi (0, 3, 0) , (83, 1, 10), (83, 1, 11)

1) op. su cilindro:	0	settore:	0	faccia :	3				
inizio:	10	seek:	(50-0)*0,5	rotazione:	3	percorrenza:	0,1	fine:	38,1
2) op. su cilindro:	83	settore:	10	faccia :	1				
inizio:	38,1	seek:	(83-0)*0,5	rotazione:	3	percorrenza:	0,1	fine:	82,7
3) op. su cilindro:	83	settore:	11	faccia :	1				
inizio:	82,7	seek:	0	rotazione:	0	percorrenza:	0,1	fine:	82,8

Disco di indice 2: al tempo t+5 sono pendenti comandi per blocchi (0, 3, 0) , (83, 1, 10).

1) op. su cilindro:	83	settore:	10	faccia :	1				
inizio:	5	seek:	(83-40)*0,5	rotazione:	3	percorrenza:	0,1	fine:	29,6
2) op. su cilindro:	0	settore:	0	faccia :	3				
inizio:	29,6	seek:	(83-0)*0,5	rotazione:	3	percorrenza:	0,1	fine:	74,2

SISTEMI OPERATIVI, CORSI A e B Prova del 16/12/2009 CORSO |A| |B|

ESERCIZIO 5 (4 punti)

Si consideri un File System FAT ospitato da un disco con *NCilindri= 128*, *NFacce= 4* e *NSettori= 128,* con blocchi di *1 kByte.* Nel disco, la copia permanente della FAT occupa il numero necessario di blocchi a partire da quello di indice 0, e i blocchi rimanenti sono riservati ai blocchi dati del File System (si ignorano il blocco di *boot* e gli ulteriori blocchi riservati a dati di controllo, come la *bitmap*). Gli elementi della FAT sono in corrispondenza uno a uno con i blocchi del disco, ma solo quelli che corrispondono a blocchi dati sono significativi, mentre non sono significativi quelli corrispondenti ai blocchi occupati dalla FAT medesima. I blocchi della FAT sono caricati a domanda nella memoria principale, le cui pagine fisiche hanno la medesima dimensione dei blocchi del disco.
Su questo File System è definito il file *pippo,* che occupa 12 blocchi consecutivi a partire da quello di indice 1525 e 15 ulteriori blocchi consecutivi a partire da quello di indice 3170. Su questo file si esegue la lettura di 3.000 caratteri, quando il puntatore di lettura ha il valore 12.000.
Si chiede:

1. la capacità del disco, in blocchi e in byte;
2. la lunghezza in byte di ciascun elemento della FAT, sufficiente a indirizzare tutti i blocchi del disco;
3. il numero (intero) di blocchi occupati dalla FAT e il numero di elementi della FAT contenuti in ciascun blocco;
4. l'indice del primo blocco del disco riservato al File System come blocco dati;
5. la massima estensione del File System, in blocchi;
6. gli indici dei blocchi dati del File System ai quali si deve accedere per la lettura;
7. gli indici dei blocchi del disco ai quali si deve accedere per leggere gli elementi della FAT che contengono i puntatori ai blocchi dati da leggere, tenendo presente che l'indice del blocco 1525 è contenuto nella directory del file *pippo;*
8. il massimo numero di Page Fault che si possono verificare per l'accesso ai blocchi di cui al punto 7..

SOLUZIONE

1. Capacità del disco: $128 * 4 * 128 = 2^{16}$ blocchi → 2^{26} = 64 Mbyte
2. La lunghezza degli elementi della FAT, necessaria per indirizzare tutti i blocchi del disco, è di 16 bit → 2 byte;
3. La FAT occupa $2^{16} * 2$ byte → $2^{17} / 2^{10}$ blocchi → 128 blocchi;
   ciascun blocco contiene 512 elementi della FAT
4. Il primo blocco del disco disponibile come blocco dati ha indice 128
5. La massima estensione del File System è di $2^{16} - 128$ blocchi → 65408 blocchi
6. Il carattere 12.000 (primo carattere da leggere) occupa la posizione 12.000 mod 1024= 736 nel blocco logico 12.000 div 1024= 11, mappato nel blocco dati 1525+ 11= 1536, che è l'ultimo del primo run. Per completare la lettura si devono leggere 3000- 736= 2264 caratteri, distribuiti su $\lceil 2264/1024 \rceil = 3$ blocchi logici, mappati sui blocchi 3170, 3171 e 3172 del secondo run. Pertanto si devono leggere i blocchi dati di indice 1536, 3170, 3171e 3172;
7. Ricordando che ogni blocco del disco contiene 512 elementi della FAT e che l'elemento *i* della FAT è associato al blocco dati *i* del file, contiene il puntatore al blocco dati *i+1* ed è contenuto nel blocco *i div 512*, gli elementi della FAT ai quali si deve accedere per eseguire la lettura sono quelli di indice:
   - 1525, 1526, 1527, 1527, 1528, 1529, 1539, 1531, 1532, 1533, 1534, 1535, contenuti nel blocco 2 del disco. L'elemento 1535 contiene il puntatore al blocco dati 1536, che è il primo blocco da leggere
   - 1536, contenuto nel blocco 3 del disco. Contiene il puntatore al blocco dati 3170.
   - 3170, 3171, contenuti nel blocco 6 del disco. L'elemento 3171 contiene il puntatore al blocco 3172, che è l'ultimo blocco dati da leggere.

   In conclusione si devono leggere dal disco i blocchi 2, 3 e 6.
8. Il massimo numero di errori di pagina è 3, per la lettura dei blocchi 2, 3 e 6 nell'ipotesi che non siano già caricati in memoria.

SISTEMI OPERATIVI, CORSI A e B Prova del 16/12/2009 CORSO |A| |B|

ESERCIZIO 6 (2 punti)

In un sistema che gestisce la memoria con segmentazione paginata, l'indirizzo logico è di 32 bit, dei quali i primi 6 bit codificano l'indice di segmento e i restanti 26 bit definiscono l'offset all'interno del segmento. I 26 bit dell'offset codificano l'indice di pagina con 14 bit e l'offset all'interno della pagina con i restanti 12 bit.
Si chiede:

- il massimo numero di segmenti di un processo,
- la dimensione di una pagina (in byte);
- la massima dimensione di un segmento (in numero di pagine);
- la massima dimensione della tabella delle pagine (in numero di descrittori).

SOLUZIONE

1. Massimo numero di segmenti di un processo: 2^6 segmenti = 64 segmenti
2. Dimensione di una pagina: 2^{12}= 4KB
3. Massima dimensione di un segmento : 2^{14}= 16K pagine
4. Massimo numero di descrittori nella tabella delle pagine: 2^{14}= 16K

ESERCIZIO 7 (2 PUNTI)

Un sistema simile a Windows gestisce la memoria con paginazione a domanda utilizzando un algoritmo LRU locale. Per ogni processo, la tabella delle pagine contiene, per ogni pagina, i campi *blocco, P* (bit di presenza), *R* (bit di uso) e *DP* (contatore che registra la distanza passata approssimata della pagina). La distanza passata è aggiornata periodicamente dal processo di sistema *WSM,* che in base al valore del bit *R* azzera o incrementa il contatore *DP*.
Al tempo *t,* quando viene attivato il processo *WSM,* la configurazione della tabella delle pagine del processo *P* è mostrata in figura.
Si chiede di completare il contenuto della tabella delle pagine al tempo *t*+ *k,* quando termina l'intervento del processo *WSM.*

SOLUZIONE

Tabella delle pagine al tempo *t*				
Pagina	blocco	P	R	DP
0	-	0	-	-
1	6	1	0	2
2	7	1	1	3
3	-	0	-	-
4	25	1	0	6
5	8	1	1	5
6	-	0	-	-
7	10	1	0	4
8	15	1	1	8
9	-	0	-	-
10	20	1	0	9
11	-	0	-	-
12	30	1	1	7

Tabella delle pagine al tempo *t*+ *k*				
Pagina	Blocco	P	R	DP
0	-	0	-	-
1	6	1	0	3
2	7	1	0	0
3	-	0	-	-
4	25	1	0	7
5	8	1	0	0
6	-	0	-	-
7	10	1	0	5
8	15	1	0	0
9	-	0	-	-
10	20	1	0	10
11	-	0	-	-
12	30	1	0	0

ESERCIZIO 8 (2 punti)

Dato il seguente frammento di codice del processo P:

```
pid=fork();
if (pid>0) {
       f=open("documenti",…);
       l=write(f,"titolo"6);
} else if (pid==0) {
       f=open("documenti",…);
       l=read(f,&buf,3);
       printf(&buf);
}
```

Nell'ipotesi che tutte le chiamate di sistema abbiano successo, dire:

1. il valore dell'I/O pointer al file documenti per il processo P
2. il valore dell'I/O pointer al file documenti per il processo figlio di P
3. Cosa stampa il processo P
4. Cosa stampa il figlio del processo P

SOLUZIONE

1. Valore dell'I/O pointer al file documenti per il processo P: 6
2. Valore dell'I/O pointer al file documenti per il processo figlio di P: 3
3. Il processo P non stampa niente
4. Il figlio del processo P stampa la stringa `"tit"`

ESERCIZIO 9 (2 punti)

Quali delle seguenti strutture dati di un processo Unix possono essere modificate dalla chiamata di sistema OPEN:

- Process structure
- Tabella dei file aperti di processo
- Tabella dei file aperti di sistema
- Tabella dei file attivi

SOLUZIONE

Struttura dati: può essere modificata?	**SI/NO**
Process structure	NO
Tabella dei file aperti di processo:	SI
Tabella dei file aperti di sistema:	SI
Tabella dei file attivi:	SI

ESERCIZIO 10 (2 punti)

In un file system UNIX si consideri il file */home/caio/foto.jpg,* appartenente all'utente *caio*.
L'utente *tizio* ha nella sua home directory il file */home/tizio/fc.jpg* che è un hard link al file *foto.jpg* dell'utente *caio*.
I diritti associati alle directory *home, tizio* e *caio* e ai file *foto.jpg* e *fc.jpg* sono i seguenti:

	owner (r w x)	*group (r w x)*	*others (r w x)*
home	*1 1 1*	*1 1 1*	*1 1 1*
tizio	*1 0 1*	*1 1 1*	*0 0 0*
caio	*1 1 1*	*0 0 0*	*0 0 0*
foto.jpg	*1 0 0*	*1 1 1*	*1 0 0*
fc.jpg	*1 1 0*	*0 1 0*	*1 0 0*

Si supponga inoltre che il contatore degli hard link del file *foto.jpg* sia pari a 2.

1. Dire quali tra le operazioni di lettura, scrittura e cancellazione sono permesse all'utente *caio* sul file *fc.jpg* supponendo che *caio* e *tizio* appartengano a gruppi diversi.
2. Nel caso in cui l'utente *tizio* cancelli il file *foto.jpg,* dire quali quali tra le operazioni di lettura, scrittura e cancellazione sono permesse all'utente *caio* sul file *fc.jpg* supponendo che *caio* e *tizio* appartengano allo stesso gruppo.

SOLUZIONE

IPOTESI	LETTURA [SI/NO]	SCRITTURA [SI/NO]	CANCELLAZIONE [SI/NO]
caio e *tizio* appartengono a gruppi diversi.	NO (***)	NO (*)	NO (**)
tizio cancella il file; (****) *caio e tizio* appartengono allo stesso gruppo.	NO	SI	SI

(*) L'utente *caio* non ha diritto di scrittura sul file *fc.jpg*
(**) L'utente *caio* non ha diritto di scrittura ne` sulla directory *tizio, ne`* sul file ***fc.jpg***
(***) L'utente *caio* non ha diritto di esecuzione sulla directory *tizio* e non può accedere allo i-node del file *fc.jpg*
(****) La cancellazione del file *caio/foto.jpg* non è possibile, perché l'utente tizio non ha il diritto di scrittura sulla directory *caio*