COMPITO 1 AULA FILA POSTO

Parte A

ESERCIZIO 1 (4 punti)

Il banco di un supermercato è gestito da due commessi che possono servire due clienti per volta. I clienti vengono serviti in base all'ordine di arrivo (si dispongono in una coda FIFO). Quando un commesso termina di servire un cliente, se ci sono clienti in attesa seleziona quello che attende da più tempo e lo serve, altrimenti resta in attesa.
I clienti (indicati come C0,…,Cn)) ed i commessi (indicati come S0 e S1) sono thread di uno stesso processo, realizzati a livello kernel.
Per la gestione del banco si utilizzano i semafori:

- `Coda`, inizializzato al valore 0,
- `Mutex`, inizializzato a 1,
- `InizioServizio[2]`, array di due semafori inizializzati a 0,
- `TermineServizio[2]`, array di due semafori inizializzati a 0,

Un cliente che arriva al banco esegue la seguente procedura:

```
ArrivoAlBanco();
AttendeServizio();
```

dove:

```
void ArrivoAlBanco() {
     wait(&Coda);

     wait(&mutex);
     S = <Seleziona un Commesso Libero tramite l'array libero[]>;
     libero[S]=False;
     Signal(&mutex);
}

void AttendeServizio() {
     Signal(&InizioServizio[S]);
     Wait (&TermineServizio[S]);
}
```

Il commesso di indice S (S=0,1) eseguono la seguente procedura:

```
while (True) {
     SelezionaCliente(S);
     ServeCliente(S);
}
```

Dove:

```
void SelezionaCliente(S) {
     Wait(&mutex);
     libero[S]=True;
     Signal(&mutex);
     Signal(&Coda);
}

void ServeCliente(S) {
     Wait(&InizioServizio[S]);
     <serve il cliente>
     Signal (&TermineServizio[S]);
}
```

Supponendo che (a partire dal tempo 0) si sussegua la seguente sequenza di eventi

Tempo 1: C1, C2,C3 (in questo ordine) eseguono `ArrivoAlBanco`
Tempo 2: S0 esegue `SelezionaCliente`
Tempo 3: C4 e C5 (in questo ordine) eseguono `ArrivoAlBanco` e S1 esegue (e completa) `SelezionaCliente`
Tempo 4: S0 esegue `ServeCliente`
Tempo 5: C1 esegue `AttendeServizio`
Tempo 6: S0 conclude `ServeCliente` e C2 esegue `AttendeServizio`

qual è lo stato raggiunto negli istanti di tempo 1,2,3,4,5 e 6?

SOLUZIONE

	Valore di	Clienti sospesi	Serventi sospesi su	Clienti sospesi su

	Coda	su Coda	InizioServizio[]	TermineServizio[]
1	0	C1, C2,C3		
2	0	C2,C3		
3	0	C3,C4,C5		
4	0	C3,C4,C5	S0	
5	0	C3,C4,C5		C1
6	0	C3,C4,C5		C2

ESERCIZIO 2 (4 punti)

Un sistema con processi A, B, C, D, E e risorse dei tipi R1, R2, R3, R4, rispettivamente di molteplicità [3, 2, 4, 5], ha raggiunto lo stato mostrato nelle tabelle seguenti.

Assegnazione attuale				
	R1	R2	R3	R4
A			1	
B			1	1
C	1			1
D	1		1	1
E				1

Esigenza residua (esclusa l'assegnazione attuale)				
	R1	R2	R3	R4
A	3	2	2	3
B	3	2	3	3
C	2	1	2	1
D	1	1	1	1
E	1	2	3	3

Molteplicità			
R1	R2	R3	R4
3	2	4	5

Disponibilità			
1	2	1	1

Dire se l'assegnazione di una risorsa di tipo R2 al processo E può essere ammessa da parte del sistema (i.e. se un'eventuale assegnazione manterrebbe il sistema in uno stato sicuro).

SOLUZIONE

Stato raggiunto dopo l'assegnazione di un'istanza di R2 al processo E:

Assegnazione attuale				
	R1	R2	R3	R4
A			1	
B			1	1
C	1			1
D	1		1	1
E		1		1

Esigenza residua (esclusa l'assegnazione attuale)				
	R1	R2	R3	R4
A	3	2	2	3
B	3	2	3	3
C	2	1	2	1
D	1	1	1	1
E	1	1	3	3

Molteplicità			
R1	R2	R3	R4
3	2	4	5

Disponibilità			
1	1	1	1

Per la verifica dello stato sicuro:

1) Il processo D può terminare
   La disponibilità di {R1, R2, R3, R4} diviene { 2 , 1 , 2 , 2 }
2) Il processo C può terminare
   La disponibilità di {R1, R2, R3, R4} diviene { 3 , 1 , 2 , 3 }
3) Il processo A può terminare NO
   La disponibilità di {R1, R2, R3, R4} diviene { … , … , … , … }
4) Il processo B può terminare NO
   La disponibilità di {R1, R2, R3, R4} diviene { … , … , … , … }
5) Il processo E può terminare NO
   La disponibilità di {R1, R2, R3, R4} diviene { … , … , … , … }

Di conseguenza: stato sicuro? NO

Il sistema può concedere un'istanza di R2 al processo E? NO

ESERCIZIO 3 (3 punti)

In un sistema vengono generati 6 processi (A,B,C,D,E), con i tempi di arrivo e le durate (in millisecondi) sotto specificate:

Processo	Durata	Tempo di arrivo
A	15	0
B	30	7
C	5	13
D	10	16
E	10	27
F	20	28

Tutti i processi avanzano senza mai sospendersi.
Calcolare il tempo di completamento di ogni processo (definito come differenza tra il tempo di completamento dell'esecuzione e il tempo di arrivo nel sistema) con le seguenti politiche di scheduling:

1. Politica Shortest Job First (tra i processi pronti manda in esecuzione il più breve, non prevede prerilascio)
2. Politica Round-Robin con quanto di tempo pari a 5 msec.

In entrambi i casi si ignori il tempo di commutazione di contesto.

Soluzione

1. Politica SJF:

PROCESSO	INIZIA ESECUZIONE	TERMINA ESECUZIONE	TEMPO DI COMPLETAMENTO
A	0	15	15
B	60	90	83
C	15	20	7
D	20	30	14
E	30	40	13
F	40	60	32

2. Politica Round Robin:

t=	0-4	5-9	10-14	15-19	20-24	25-29	30-34	35-39	40-44	45-49	50-54	55-59	60-64	65-69	70-74	75-79	80-84	85-89
A	*	*		*														
B			*			*				*				*		*		*
C					*													
D							*				*							
E								*				*						
F									*				*		*		*	

Tempo di completamento:

A : 20; B : 90; C : 25; D : 55; E : 60; F : 85;

ESERCIZIO 4 (2 punti)

Si consideri un sistema operativo con thread realizzati a livello utente. Quali dei seguenti dati sono contenuti nel descrittore di processo e quali nel descrittore di thread?

Dato	Descr. Processo	Descr Thread
Immagine del contatore di programma	SI	SI
Puntatore alle aree dati in memoria	SI	NO
Immagine dei registri del processore	SI	SI
Stato (pronto/esecuzione/bloccato)	SI	SI
Puntatore alla tabella dei file aperti	SI	NO
Puntatore allo stack	SI	SI

ESERCIZIO 5 (2 punti)

Si considerino due processi Unix A e B che comunicano tramite una pipe con un buffer di dimensione 2000 byte, e supponiamo che A abbia accesso alla pipe in scrittura mentre B abbia accesso in lettura.
Dire in quali delle seguenti circostanze il processo B si blocca:

1. Il buffer della pipe è vuoto e il processo B effettua una lettura dalla pipe
2. Il buffer della pipe è vuoto e il processo A scrive un messaggio di dimensione 100 byte
3. Il buffer della pipe è pieno e il processo A scrive un messaggio di dimensione 1000 byte
4. Il buffer della pipe è pieno e il processo B effettua una lettura dalla pipe

Soluzione

1. ……SI……
2. ……NO……
3. ……NO……
4. ……NO……

Parte B

ESERCIZIO 6 (4 PUNTI)

Un disco con 4 facce, 50 settori per traccia e 200 cilindri ha un tempo di seek (proporzionale al numero di cilindri attraversati) pari a 0,5 ms per ogni cilindro. Il periodo di rotazione è di 10 msec: di conseguenza il tempo impiegato per percorrere un settore è 0,2 msec.
Inoltre si fanno le seguenti ipotesi :

- il tempo necessario per raggiungere il settore desiderato dopo un'operazione di seek è sempre quello massimo, pari a 10 msec.
- il disco dispone di un buffer della capacità di un settore.

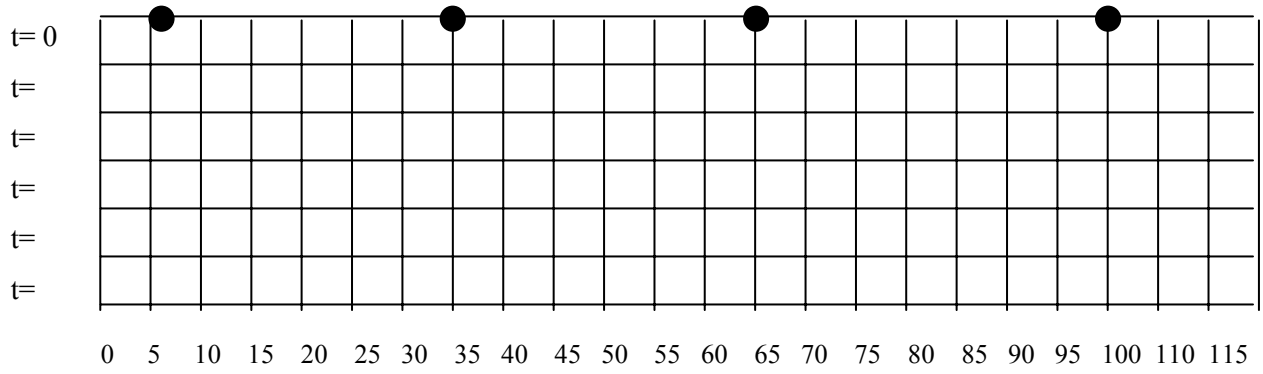
Al tempo 0 termina l'esecuzione dei comandi relativi al cilindro 170 e sono pendenti le seguenti richieste di lettura o scrittura:

- cilindro 150 : settori 20 e 19 della faccia 0
- cilindro 190 : settore 44 della faccia 1
- cilindro 165 : settori 30 della faccia 0 e 35 della faccia 1

Successivamente arrivano i seguenti comandi:

- al tempo 20: cilindro 195 settore 7 della faccia 4.
- al tempo 50: cilindro 150 settore 12 della faccia 2.
- al tempo 80: cilindro 180 settore 40 della faccia 2.

Lo scheduling è effettuato con politica SSF. Calcolare il tempo necessario per eseguire tutte le operazioni. Il tempo di esecuzione di ogni operazione è uguale alla somma dell'eventuale tempo di *seek,* del ritardo rotazionale (tempo necessario per raggiungere il settore indirizzato) e del tempo di percorrenza del settore indirizzato. Quando si raggiunge un cilindro, i comandi pendenti devono essere eseguiti nell'ordine in cui sono elencati.
Per il ritardo rotazionale dopo un'operazione di *seek* si assume sempre il valore di caso peggiore, pari a un periodo di rotazione.



SOLUZIONE

op. su cilindro:	165	;	settore:	30							
inizio:	0	;	seek:	2,5	;	rotazione:	10	;	percorrenza:	0,2	; fine: 12,7
op. su cilindro:	165	;	settore:	35							
inizio:	12,7	;	seek:	0	;	rotazione:	1	;	percorrenza:	0,2	; fine: 13,9
op. su cilindro:	150	;	settore:	20							
inizio:	13,9	;	seek:	7,5	;	rotazione:	10	;	percorrenza:	0,2	; fine: 31,6
op. su cilindro:	150	;	settore:	19							
inizio:	31,6	;	seek:	0	;	rotazione:	9,8	;	percorrenza:	0,2	; fine: 41,6
op. su cilindro:	190	;	settore:	44							
inizio:	41,6	;	seek:	20	;	rotazione:	10	;	percorrenza:	0,2	; fine: 71,8
op. su cilindro:	195	;	settore:	7							
inizio:	71,8	;	seek:	2,5	;	rotazione:	10	;	percorrenza:	0,2	; fine: 84,5
op. su cilindro:	180	;	settore:	40							
inizio:	84,5	;	seek:	7,5	;	rotazione:	10	;	percorrenza:	0,2	; fine: 102,2
op. su cilindro:	150	;	settore:	12							
inizio:	102,2	;	seek:	15	;	rotazione:	10	;	percorrenza:	0,2	; fine: 127,4

ESERCIZIO 7 (4 punti)

In un sistema che gestisce la memoria con paginazione, sono presenti i processi A, B e C.
Le tabelle delle pagine dei tre processi sono le seguenti (con notazione evidente; in parentesi è replicato il tempo al quale è avvenuto l'ultimo riferimento)

Pagina	Blocco
0	-
1	6 (2)
2	-
3	-
4	-
5	8 (9)
6	-
7	10 (4)

Processo A

Pagina	Blocco
0	7 (1)
1	-
2	-
3	-
4	5 (10)
5	-
6	-
7	11 (3)

Processo B

Pagina	Blocco
0	9 (6)
1	1 (0)
2	-
3	12 (3)
4	2 (5)
5	-
6	-
7	-

Processo C

Per la gestione della memoria si utilizza un algoritmo di sostituzione LRU locale. Il working set (inteso come numero di blocchi a disposizione del processo, anche se momentaneamente non occupati) inizialmente assegnato ai processi è il seguente:

- processo A: 6,8,10
- processo B: 7,5,11,4
- processo C: 9,1,12,2

Quali pagine vengono scaricate dalla memoria e caricate in memoria se si verificano in sequenza i seguenti eventi:

- tempo 11: il processo A riferisce la pagina 2 ;
- tempo 12: il working set del processo A viene aumentato di un blocco (gli viene assegnato il blocco 3) ;
- tempo 13: il processo A riferisce la pagina 6 ;
- tempo 14: il processo A viene deschedulato e passa in esecuzione il processo B ;
- tempo 15: il processo B riferisce la pagina 6 ;
- tempo 16: il processo B riferisce la pagina 1 ;
- tempo 17: il processo B termina e passa in esecuzione il processo C ;
- tempo 18: il processo C riferisce la pagina 5 ;
- tempo 19: processo C riferisce la pagina 1 ;
- tempo 20: processo C riferisce la pagina 3 ;

SOLUZIONE

tempo 11: processo: A , pagina logica rimossa : 1 ; pagina logica 2 caricata sul blocco 6 tempo di riferimento: 11 ;
tempo 12: processo: - , pagina logica rimossa : - ; pagina logica - caricata sul blocco - tempo di riferimento: - ;
tempo 13: processo: A , pagina logica rimossa : - ; pagina logica 6 caricata sul blocco 3 tempo di riferimento: 13 ;
tempo 14: processo: - , pagina logica rimossa : - ; pagina logica - caricata sul blocco - tempo di riferimento: - ;
tempo 15: processo: B , pagina logica rimossa : - ; pagina logica 6 caricata sul blocco 4 tempo di riferimento: 15
tempo 16: processo: B , pagina logica rimossa : 0 ; pagina logica 1 caricata sul blocco 7 tempo di riferimento: 16
tempo 17: processo: - , pagina logica rimossa : - ; pagina logica - caricata sul blocco - tempo di riferimento: -
tempo 18: processo: C , pagina logica rimossa : 1 ; pagina logica 5 caricata sul blocco 1 tempo di riferimento: 18
tempo 19: processo: C , pagina logica rimossa : 3 ; pagina logica 1 caricata sul blocco 12 tempo di riferimento: 19
tempo 20: processo: C , pagina logica rimossa : 4 ; pagina logica 3 caricata sul blocco 2 tempo di riferimento: 20

Tabella delle pagine al tempo 21:

Pagina	Blocco
0	-
1	-
2	6 (11)
3	-
4	-
5	8 (9)
6	3 (13)
7	10 (4)

Processo A

Pagina	Blocco
0	-
1	7 (16)
2	-
3	-
4	5 (10)
5	-
6	4 (15)
7	11 (3)

Processo B

Pagina	Blocco
0	9 (6)
1	12 (19)
2	-
3	2 (20)
4	-
5	1 (18)
6	-
7	-

Processo C

Esercizio 8 (3 punti)

In un file system di tipo FAT in cui i blocchi fisici hanno dimensione 1KB (1024 byte) è presente il file A il cui blocco iniziale è 11.
Dato il seguente frammento di FAT:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
-		7	5		-1		8	9	13		23		3								2	21	22	

dire in quali blocchi fisici sono memorizzati i seguenti bytes del file A:

a- byte 5000 b- byte 400 c- byte 3990 d- byte 4096 e- 8192 f- 10240

Soluzione

a- Blocco logico: 4 Blocco fisico: 2
b- Blocco logico: 0 Blocco fisico: 11
c- Blocco logico: 3 Blocco fisico: 21
d- Blocco logico: 4 Blocco fisico: 2
e- Blocco logico: 8 Blocco fisico: 13
f- Blocco logico: 10 Blocco fisico: 5

Esercizio 9 (2 punti)

In un file system UNIX si consideri il file */home/tizio/personali/dati.*
I diritti associati alle directory *home e tizio* e al file *dati* sono i seguenti (le terne sono riferite, nell'ordine, a *user*, *group* e *others* e ogni terna riporta, nell'ordine, i diritti *r, w, x*):

tizio	101	101	101
personali	101	100	001
dati	010	101	100

Quali tra le operazioni di lettura, scrittura e cancellazione possono essere eseguite sul file *dati* da:

1. il proprietario della directory *tizio*
2. un utente che appartiene allo stesso gruppo del proprietario della directory *tizio*;
3. un utente che NON appartiene allo stesso gruppo del proprietario della directory *tizio*;

Soluzione

1. Lettura: NO ; Scrittura: SI ; Cancellazione: NO ;
2. Lettura: SI ; Scrittura: NO ; Cancellazione: NO ;
3. Lettura: SI ; Scrittura: NO ; Cancellazione: NO ;

Esercizio 10 (2 punti)

Un disco RAID di livello 4 è composto da 6 dischi fisici, numerati da 0 a 5. Le strip corrispondono a blocchi.
Il disco 5 è ridondante e il suo blocco di indice *i* contiene la parità dei blocchi di indice i dei dischi non ridondanti, cioè dei dischi 0, 1, 2, 3 e 4.
I blocchi di indice 6 dei blocchi non ridondanti contendono rispettivamente:
Disco 0: 1 1 0 0 1 0 1 0
Disco 1: 0 1 1 1 1 1 0 0
Disco 2: 0 0 0 0 0 0 0 0
Disco 3: 1 1 0 0 1 0 1 0
Disco 4: 0 1 1 0 0 1 1 1

Si chiede:

1) il contenuto del blocco di indice 6 del disco ridondante
2) quali blocchi è necessario leggere per scrivere 1 1 1 1 1 1 1 1 nel blocco di indice 6 del disco 0
3) quali blocchi è necessario modificare per scrivere 1 1 1 1 1 1 1 1 nel blocco di indice 6 del disco 0

Soluzione

1) contenuto del blocco di indice 6 del disco ridondante : 0 0 0 1 1 0 1 1
2) si leggono i blocchi: di indice 6 dei dischi 0 e 5
3) si modificano i blocchi: di indice 6 dei dischi 0 e 5