Caselle riservate

Ex. 1	
Ex. 2	
Ex. 3	
Ex. 4	
Ex. 5	
Ex. 6	
Tot.	

Sistemi Operativi

Compito d'esame

01 Febbraio 2019

Matricola _____ Cognome _____ Nome _____

Docente: ◯ Quer ◯ Sterpone

Non si possono consultare testi, appunti o calcolatrici a parte i formulari distribuiti dal docente. Risolvere gli esercizi negli spazi riservati. Fogli aggiuntivi sono permessi solo quando strettamente necessari. Riportare i passaggi principali.
Durata della prova: 100 **minuti.**

1. Si supponga che il disco rigido di un piccolo sistema embedded sia costituito da 24 blocchi di 1 MByte, che tali blocchi siano numerati da 0 a 23, che il sistema operativo mantenga traccia dei blocchi liberi (occupati) indicandoli in un vettore con il valore 0 (1), e che la situazione attuale del disco sia rappresentata dal seguente vettore:

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

   Con riferimento alle metodologie di allocazione di file contigua, concatenata, FAT e indicizzata, indicare come possono essere allocati in sequenza i file `File1`, `File2` e `File3` di dimensione uguale a 2.4, 1.6 e 3.9 Mbyte, rispettivamente.

   Riportare schematicamente per ogni strategia le informazioni memorizzate nella directory entry ed i relativi puntatori.

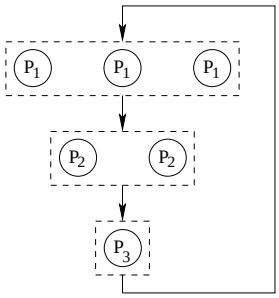
2. Con riferimento alle soluzioni hardware al problema della sincronizzazione, si riportino le soluzioni mediante le procedure di `testAndSet` e di `swap`. Se ne illustrino le principali differenze (vantaggi e svantaggi) rispetto alle tecniche software, nonchè la loro relazione con il problema della "starvation".

   Si indichi inoltre che cosa si intende per "spin-lock" e per "priority inversion".

3. Un programma concorrente è costituito da 3 processi (P_1, P_2, P_3) **ciclici**, **non** ricorsivi, che **non** si richiamano a vicenda e di cui sono presenti 3, 2, e 1 istanza, rispettivamente. I processi sono tali per cui:

   - All'inizio sono eseguite le 3 istanze del processo P_1 in parallelo.
   - Al termine dell'ultima istanza di P_1, sono eseguite le 2 istanze del processo P_2 in parallelo.
   - Al termine dell'ultima istanza di P_2, è eseguita l'unica istanza del processo P_3.
   - Al termine dell'esecuzione dell'istanza di P_3, il procedimento riprende dall'inizio con l'esecuzione delle tre istanze di P_1.

   La figura successiva mostra la relazione temporale tra i processi.



   Si scriva il programma (in pseudo-codice) illustrandone i meccanismi di sincronizzazione utilizzando il minimo numero di semafori possibile. Si utilizzino dei contatori per sincronizzare i processi come richiesto.

4. Un file ha il seguente formato:

```
# prog1
f1.c
func1.c
main1.c
my_func.c
END
# prog3
main2.c
func2_2.c
my_func.c
END
...
```

Si implementi uno script BASH in grado di:

- Ricevere il nome di un file sulla riga di comando. Tale file ha il formato precedentemente indicato.
- Compilare i file i cui nomi sono indicati sulle righe che non iniziano con il carattere `#`, generando l'eseguibile il cui nome è indicato sulla riga che inizia con il carattere `#` e che precede i nomi dei file sorgente da compilare. Le righe contenenti la stringa `END` indicano la terminazione dell'elenco dei file sorgente da compilare.

Lo script deve inoltre generare un direttorio di nome `log` (nel caso non esista già) e copiare in tale direttorio l'output di tutti i comandi di compilazione ciascuno in un file con lo stesso nome dell'eseguibile generato e estensione `.log`.

Per esempio eseguendo lo script sul file sorgente precedente, si dovrebbero generare i seguenti comandi:

```
gcc -Wall -o prog1 f1.c func1.c main.c my_func.c
gcc -Wall -o prog3 main2.c func2_2.c my_func.c
...
```

memorizzando il loro output nei file `./log/prog1.log` e `./log/prog3.log`, rispettivamente.

5. **Per i candidati iscritti al corso nell'anno accademico 2018–2019.**
   In algebra lineare, la moltiplicazione di matrici è l'operazione che produce una nuova matrice C effettuando il prodotto righe per colonne di due matrici date A e B. Più in dettaglio se A ha dimensione $[r, x]$ e B ha dimensione $[x, c]$, allora C avrà dimensione $[r, c]$ e ciascuno dei suoi elementi di posizione (i, j) sarà calcolabile come:

$$C[i][j] = \sum_{k=0}^{x-1} A[i][k] \cdot B[k][j]$$

   Si scriva una funzione multi-thread

```
void mat_mul (int **A, int **B, int r, int x, int c, int **C);
```

   in grado di calcolare la matrice prodotto C, eseguendo un thread per ciascuno dei suoi elementi. Ciascun thread si occuperà di calcolare il valore dell'elemento stesso, effettuando il prodotto righe per colonne precedentemente indicato. Si definisca la struttura dati necessaria all'esecuzione dei thread.

Per i candidati iscritti al corso prima dell'anno accademico 2018–2019.
Il cifrario di Cesare, uno dei più antichi algoritmi crittografici, prevede che ogni lettera di un testo in chiaro venga sostituita con una lettera che si trova un certo numero di posizioni dopo nell'alfabeto. Per generalizzazione ogni lettera può essere sostituita con una lettera equivalente.

Si supponga le equivalenze tra lettere vengano riportate in un file. La prima riga del file contiene le lettere originali, mentre la seconda specifica le lettere corrispondenti. Il seguente è un esempio corretto:

```
a b c d e f g h i j k l m n o p q r s t u v w x y z
d e f g h i j k l m n o p q r s t u v w x y z a b c
```

Si scriva uno script AWK in grado di:

- Ricevere il nome di tre file sulla riga di comando. Il primo file contiene il codice. Il secondo memorizza un testo di formato e lunghezza indefinita che occorre cifrare. Il terzo file, che lo script deve generare, conterrà il testo cifrato.
- Crittografi il testo specificato e memorizzi tale testo nel terzo file.

Ad esempio, con la codifica precedentemente indicata, il file

```
attaccare gli irriducibili galli
alla ora sesta.
```

genera il seguente file:

```
dwwdffduh jol luulgxflelol jdool
dood rud vhvwd.
```

6. Si consideri il seguente insieme di processi:

Processo	Tempo arrivo	Burst Time	Priorità
P_0	0	22	2
P_1	0	26	3
P_2	5	19	1
P_3	11	17	4
P_4	13	18	5

Rappresentare mediante diagramma di Gantt l'esecuzione di tali processi utilizzando gli algoritmi di scheduling PS (Priority Scheduling), RR (Round Robin) e SRTF (Shortest Remaining Time First). Calcolare il tempo di attesa medio per ciascun processo e quello globale. Si consideri un quantum temporale di 10 unità di tempo.

Si illustrino quali altre metriche di valutazioni sarebbe possibile utilizzare al fine di confrontare gli algoritmi di scheduling precedentemente indicati.