Sistemi Operativi
I Prova in Itinere

26 Aprile 2022

Esercizio 1 (8 punti)
Si consideri il seguente segmento di codice concorrente:

```
int c = 4;
pid_t pid = fork();
if (pid == 0) {
    c += 4;
} else {
    pid = fork();
    c += 8;
    if (pid) {
        c += 8;
    }
}
fork();
printf("%d\n", c);
```

1) Compreso il processo iniziale, che si origina dall'esecuzione del programma, quanti processi vengono creati a seguito delle invocazioni alla `fork()`?
2) Si descriva l'albero dei processi e si dica quali sono gli output prodotti.

Esercizio 2 (8 punti)
Si considerino i processi A e B con tempi di arrivo e burst di CPU ed I/O specificati nella seguente tabella:

Processo	Tempo di arrivo	CPU–1	I/O–1	CPU–2	I/O–2	CPU–3
A	0	5	10	3	5	4
B	4	6	12	5	10	4

Si assuma che il tempo di context switch sia trascurabile e che vi sia un singolo dispositivo di I/O, che viene gestito con politica FCFS. Tracciare i diagrammi di Gantt per CPU e I/O, calcolare il tempo medio di attesa e di turnaround e valutare l'utilizzo percentuale della CPU con scheduling RR con quanto uguale a 4.

Esercizio 3 (8 punti)
Si consideri il seguente insieme di task periodici con relativi tempi di arrivo, tempi di esecuzione e di scadenza:

Task	Tempo di arrivo (msec)	Esecuzione (msec)	Scadenza (msec)	Periodo (msec)
T_1	0	1	4	4
T_2	0	2	6	6
T_3	0	3	8	8

Si valuti preventivamente se lo scheduling è ammissibile. In caso affermativo, si tracci il diagramma di Gantt in base alla politica di scheduling EDF.

Esercizio 4 (6 punti)
1) Siano:

```
y = 0; r = 2; x = 3; z = 0;
```

i dati condivisi dai due processi concorrenti `A` e `B` tali che

```
A:  r=r+1;
    x=x+1;
    print(x);
    print(r);

B:  y=x;
    z=z+1;
    print(z);
```

Quali e quanti semafori vanno aggiunti per garantire che il valore di `y` sia 4? Inizializzati come?

2) Si considerino i processi A e B seguenti, con i relativi dati condivisi:

```
semaphore mutex = 1; semaphore time_a = 2; semaphore time_b = 0;

A:      while(1){                       B:      while(1){
            wait(time_a);                           wait(time_b);
            wait(mutex);                            wait(mutex);
            <SCA>                                   <SCB>
            signal(mutex);                          signal(mutex);
            signal(time_b);                         signal(time_a);
        }                                       }
```

L'esecuzione concorrente di A e B produce una sequenza infinita di <SCA> e <SCB>. Qual è l'unica sequenza possibile fra le seguenti? Giustificare la risposta data, mostrando come le altre non siano realizzabili.

a) SCA, SCA, SCB, SCA, SCA, SCB, SCA, SCA, SCB...
b) SCA, SCB, SCA, SCA, SCB, SCA, SCB, SCA, SCA,...
c) SCA, SCB, SCA, SCB, SCA, SCB, SCA, SCB, SCA,...
d) SCA, SCA, SCB, SCB, SCA, SCB, SCB, SCA, SCA,...

NO

Esercizio 5 (5 punti)

Si assuma un sistema di calcolo con quattro processi, P_1, P_2, P_3 e P_4 in esecuzione concorrente e tre tipi di risorse, R_1, R_2 e R_3. Il numero totale di risorse disponibili è dato dal vettore [5,8,16], mentre le matrici che descrivono le richieste massime e l'allocazione di risorse attuale per ciascun processo sono rispettivamente:

$$Max = \begin{bmatrix} 4 & 1 & 4 \\ 3 & 1 & 4 \\ 5 & 7 & 13 \\ 1 & 1 & 6 \end{bmatrix} \qquad Allocation = \begin{bmatrix} 0 & 1 & 4 \\ 2 & 0 & 1 \\ 1 & 2 & 1 \\ 1 & 0 & 3 \end{bmatrix}$$

a) Si determini se lo stato corrente è sicuro; SI
b) Si valuti se un'ulteriore richiesta del processo P_1 per un'istanza della risorsa di tipo R_1 mantiene il sistema in stato sicuro;
c) Qualunque sia lo stato raggiunto (in dipendenza dal punto precedente), si valuti se un'ulteriore richiesta del processo P_3 per sei istanze della risorsa R_3 mantiene il sistema in stato sicuro.