

UNIVERSITA' DEGLI STUDI DI BOLOGNA - CORSO DI LAUREA IN INFORMATICA
PROVA SCRITTA DI SISTEMI OPERATIVI
ANNO ACCADEMICO 2020/2021
14 febbraio 2022

Esercizio -1: Essere iscritti su AlmaEsami per svolgere questa prova.

Esercizio 0: Scrivere correttamente nome, cognome, matricola e posizione in tutti i fogli prima di svolgere ogni altro esercizio. Scrivere esclusivamente a penna senza abrasioni. E' vietato l'uso delle penne cancellabili, della matita, dei coprenti bianchi per la correzione (bianchetto) e la scrittura in colore rosso (riservato alla correzione). Il compito e' formato da tre fogli, sei facciate compresa questa. Le soluzioni che si vogliono sottoporre per la correzione devono essere scritte negli spazi bianchi di questi fogli. Non verranno corretti altri supporti. E' obbligatorio consegnare il compito, e' possibile chiedere che esso non venga valutato scrivendo "NON VALUTARE" in modo ben visibile nella prima facciata. Per svolgere questo compito occorre solo una penna e un documento di identità valido. La consultazione o anche solo la disponibilità di altro materiale comporterà l'annullamento del compito (verrà automaticamente valutato gravemente insufficiente).

Esercizio c.1: Scrivere il monitor `semdata` che implementi un semaforo con dato. Questa astrazione prevede due operazioni:

```
datatype dP(void);  
void dV(datatype data);
```

Non è previsto assegnamento di valore iniziale nel costruttore, l'invariante è lo stesso dei semafori (con `init = 0`): `ndP <= ndV` (dove `ndP` e `ndV` rappresentano rispettivamente il numero di operazioni `dP` e `dV` completate. I dati passati come parametro alla `dV` devono essere memorizzati in ordine LIFO. L'operazione `nP` restituisce il valore più recente fra quelli memorizzati (e lo cancella dalla struttura dati).

Esercizio c.2: Dato un servizio di message passing asincrono implementare un servizio di message passing sincrono con selezione ordinata che ha la seguente interfaccia:

```
void snd(msgtype msg, pid_t dest);  
msgtype snrcv(pid_t sender, int n);
```

La funzione `snrcv` deve restituire l'`n`-mo messaggio proveniente dal mittente specificato (che può essere `any`). Se `n == 0` restituisce l'ultimo messaggio. Esempi:

`m = snrcv(tizio, 1)`: restituisce il primo messaggio da tizio (attende se non ve ne sono)

`m = snrcv(ANY, 42)`: restituisce il 42-mo messaggio da chiunque (attende se ci sono meno di 42 messaggi in attesa di essere ricevuti)

`m = snrcv(caio, 0)`: restituisce l'ultimo messaggio ricevuto da Caio (attende se non ci sono messaggi pendenti da Caio)

`m = snrcv(ANY, 0)`: restituisce l'ultimo messaggio ricevuto, indipendentemente dal mittente.

Esercizio g.1: Considerare i seguenti processi gestiti mediante uno scheduler preemptive a priorità statica su una macchina biprocessore SMP:

P1: cpu 4 ms; I-O 4 ms; cpu 2 ms

P2: cpu 2 ms; I-O 4 ms; cpu 5 ms

P3: cpu 5 ms; I-O 3 ms; cpu 3 ms

P4: cpu 10 ms; I-O 1 ms

l'Input-Output avviene su un'unica unità. Per il processo P1 ha priorità minima seguito da P2, P3 e P4 in sequenza crescente, le richieste di I/O sono gestite in ordine FIFO. Calcolare il tempo necessario a completare l'esecuzione dei 4 processi. Indicare con chiarezza i punti dello schedule nei quali avviene la preemption.

Esercizio g.2: rispondere alle seguenti domande (motivando opportunamente le risposte):

a) Perché viene usata la paginazione per implementare la memoria virtuale?

b) L'algoritmo del banchiere dato uno stato di allocazione delle risorse restituisce un valore binario: safe o non safe.

In quali casi il sistema operativo esegue l'algoritmo del banchiere? Cosa succede se il risultato è safe e cosa se il risultato è non-safe?

c) Fornire esempi di file system con allocazione contigua, e spiegare perché sarebbe inefficiente usare altri metodi di allocazione nei casi d'uso tipici di questi file system.

d) perché l'invenzione degli interrupt ha reso i sistemi operativi più efficienti?