- Che cos'è un sistema time-sharing? Si può avere time-sharing in un sistema con singola CPU?

  Risposta: Un sistema time-sharing è una variante della multiprogrammazione in cui il/i processore/i esegue più processi commutando le loro esecuzioni con una frequenza sufficientemente elevata da permettere a ciascun utente di interagire con il proprio programma durante la sua esecuzione. Il time-sharing si può avere anche in un sistema con una singola CPU.

- Quali sono gli svantaggi di un sistema operativo basato su thread rispetto ad uno basato su processi?

  Risposta: Gli svantaggi di un sistema operativo basato su thread rispetto ad uno basato su processi si riassumono sostanzialmente in una maggiore complessità di progettazione e programmazione. Infatti i processi devono essere "pensati" ed organizzati in attività concorrenti, c'è un minor *information hiding* in quanto lo spazio di memoria fra i thread di uno stesso processo è condiviso. Inoltre bisogna porre estrema cura nella sincronizzazione dei thread e nel loro scheduling (che spesso è demandato all'utente). I thread infine risultano particolarmente inadatti a situazioni in cui è necessario proteggere i dati.

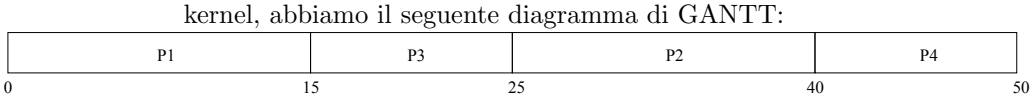
- In coda ready arrivano i processi P_1, P_2, P_3, P_4, con CPU burst e istanti di arrivo specificati in tabella:

	arrivo	burst
P_1	0	15ms
P_2	10	15ms
P_3	15	10ms
P_4	30	10ms

  1. Se i processi terminano nell'ordine P_1, P_3, P_2, P_4, quale può essere l'algoritmo di scheduling? (Si trascuri il tempo di latenza del kernel.)
     (a) RR q=20ms
     (b) RR q=10ms
     (c) Scheduling preemptive
     (d) SRTF
     (e) Nessuno dei precedenti
  2. (*) Si calcoli il tempo di turnaround medio per i processi P_1, P_3, P_2, P_4 della tabella sopra nel caso di algoritmo SRTF.

  Risposta:

  1. b), c), d).
  2. Considerando un algoritmo di scheduling SRTF senza latenza del kernel, abbiamo il seguente diagramma di GANTT:



     Quindi il tempo di turnaround medio è $\frac{15+(40-10)+(25-15)+(50-30)}{4} = 75/4 = 18,75ms$.

- Si consideri la seguente situazione, dove P_0, P_1, P_2, P_3, P_4 sono cinque processi in esecuzione, C è la matrice delle risorse correntemente allocate, Max è la matrice del numero massimo di risorse assegnabili ad ogni processo e A è il vettore delle risorse disponibili:

	C				Max			
	A	B	C	D	A	B	C	D
P_0	0	2	0	2	0	3	1	2
P_1	0	0	0	0	2	7	5	0
P_2	2	3	5	4	2	3	7	6
P_3	0	4	3	2	0	4	5	2
P_4	0	0	1	5	0	6	5	5

Available (A)

A	B	C	D
1	5	3	0

1. Calcolare la matrice R delle richieste.
2. Il sistema è in uno stato sicuro (safe)?

Risposta:

1. La matrice R delle richieste è data dalla differenza $Max - C$:

R			
A	B	C	D
0	1	1	0
2	7	5	0
0	0	2	2
0	0	2	0
0	6	4	0

2. Sì il sistema è in uno stato sicuro in quanto esiste la sequenza sicura $\langle P_0, P_2, P_1, P_3, P_4\rangle$. Infatti, dapprima si esegue P_0 in quanto $R_0 \leq A$ ed A diventa quindi $(1, 7, 3, 2)$. A questo punto $R_2 \leq A$ e quindi si esegue P_2 generando il nuovo valore di A: $(3, 10, 8, 6)$. Quindi si può mandare in esecuzione P_1 dato che $R_1 \leq A$ lasciando inalterato A, dato che $C_1 = (0, 0, 0, 0)$. In seguito può essere eseguito P_3 ($R_3 \leq A$) aggiornando A al valore $(3, 14, 11, 8)$. Infine viene eseguito R_4 ($C_4 \leq A$) generando il valore finale di $A = (3, 14, 12, 13)$.

- 1. Cosa si intende con l'espressione *frammentazione interna* e con l'espressione *frammentazione esterna*?
  2. (*) Supponendo di avere un sistema con quattro frame e sette pagine, addottando una politica di rimpiazzamento LRU, quanti page fault si verificheranno con la reference string seguente?

     0 1 6 2 3 2 6 1 2 3 1

     (Si assuma che i quattro frame siano inizialmente vuoti.)

Risposta:

1. Quando la memoria viene allocata in blocchi di dimensione fissata, con l'espressione frammentazione interna si intende la differenza fra la memoria assegnata ad un processo e quella effettivamente richiesta da quest'ultimo. Quando si alloca la memoria in blocchi di dimensione variabile e si caricano e si rimuovono da quest'ultima dei processi,

lo spazio libero si frammenta in piccole parti; si parla di frammentazione esterna quando lo spazio libero complessivo nella memoria è sufficiente per soddisfare una richiesta, ma non è contiguo.

2. Simuliamo il funzionamento di LRU:

	0	1	6	2	3	2	6	1	2	3	1
		0	1	6	2	3	2	6	1	2	3
			0	1	6	6	3	2	6	1	2
				0	1	1	1	3	3	6	6
					0	0	0	0	0	0	0
	P	P	P	P	P						

Si verificano quindi cinque page fault.

- Descrivere le differenze fra le seguenti modalità di I/O:
  - Programmed I/O (PIO),
  - Interrupt-driven I/O,
  - Direct Memory Access (DMA).

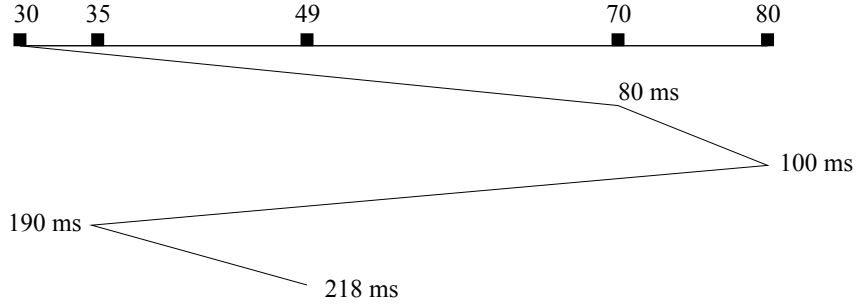
  Risposta: Mentre con la modalità Programmed I/O (I/O a interrogazione ciclica) il processore manda un comando di I/O e poi attende che l'operazione sia terminata, testando lo stato del dispositivo con un loop busy-wait (polling), con la modalità Interrupt-driven I/O, una volta inviato il comando di I/O, il processo viene sospeso fintanto che non arriva un interrupt a segnalare il completamento dell'operazione. Durante la sospensione del processo, la CPU può mandare in esecuzione altri processi o thread. Di fondamentale importanza è il vettore di interrupt che consente di selezionare la routine di gestione opportuna per ogni tipo di interrupt. Ovviamente la prima modalità è efficiente soltando nel caso in cui la velocità del dispositivo di I/O sia paragonabile a quella della CPU. La modalità DMA richiede un controller DMA e funziona in questo modo: la CPU imposta i registri del controller DMA specificando il tipo di azione di I/O, l'indirizzo di memoria ed il conteggio di byte da trasferire. Poi i dati vengono trasferiti senza più richiedere l'intervento della CPU; infatti il controller del dispositivo di I/O riceve le richieste di lettura o scrittura da parte del controller DMA a cui notifica il completamento dell'operazione una volta che ha trasferito il byte da/verso l'indirizzo di memoria corretto (specificato dal controller DMA). A questo punto il controller DMA incrementa l'indirizzo di memoria comunicandolo sul bus e decrementa il conteggio dei byte da traferire, ripetendo la richiesta di lettura o scrittura al controller del dispositivo fintanto che il conteggio dei byte non raggiungerà lo zero. Soltanto a questo punto verrà inviato un interrupt alla CPU che potrà far ripartire il processo sospeso. Siccome il controller DMA deve bloccare il bus per consentire i trasferimenti dal controller del dispositivo alla memoria, se anche la CPU ha bisogno di accedere al bus dovrà aspettare, venendo così rallentata.

- Spiegare brevemente la differenza fra servizi di rete e servizi distribuiti.

Risposta: I servizi di rete offrono ai processi le funzionalità necessarie per stabilire e gestire le comunicazioni tra i nodi di un sistema distribuito (es.: l'interfaccia fornita dalle socket). In sostanza gli utenti devono essere consapevoli della struttura del sistema e devono indirizzare esplicitamente le singole macchine. I servizi distribuiti invece sono modelli comuni (paradigmi di comunicazione) trasparenti che offrono ai processi una visione uniforme, unitaria del sistema distribuito stesso (es: file system remoto). I servizi distribuiti vanno quindi a formare il cosiddetto *middleware*, ovvero, uno strato software fra il sistema operativo e le applicazioni che uniforma la visione dell'intero sistema.

- (*) Si consideri un disco gestito con politica C-LOOK. Inizialmente la testina è posizionata sul cilindro 30, ascendente; lo spostamento ad una traccia adiacente richiede 2 ms. Al driver di tale disco arrivano richieste per i cilindri 70, 49, 35, 80, rispettivamente agli istanti 0 ms, 40 ms 50 ms, 70 ms. Si trascuri il tempo di latenza. In quale ordine vengono servite le richieste?

  Risposta: L'ordine in cui vengono servite le richieste è 70, 80, 35, 49 come illustrato dal seguente diagramma:



Il punteggio attribuito ai quesiti è il seguente: 3, 3, 3, 3, 2, 2, 2, 5, 3, 3, 4 (totale: 33).