- In un sistema operativo basato su processi tradizionali, quali sono le informazioni relative a ciascun processo necessarie per gestire il context switch? Dove sono mantenute?

  Risposta: Le informazioni relative a ciascun processo per il context switch sono: il valore del program counter e della program status word, i registri, lo stack pointer, lo stato di esecuzione. Tali informazioni vengono salvate nell'entry (process control block) relativa al processo della process table mantenuta dal sistema operativo.

- Quali algoritmi di scheduling della CPU sono adatti a sistemi in cui sono presenti sia processi time-sharing che processi real-time?

  Risposta: Nei sistemi in cui convivono processi time-sharing e processi real-time gli algoritmi di scheduling della CPU che risultano più efficaci sono quelli con code multiple in cui la coda relativa ai processi real-time è quella con maggiore priorità, mentre i processi time-sharing sono organizzati in code con priorità minori. All'interno delle varie code poi gli algoritmi di scheduling sono solitamente FCFS o RR per i processi real-time e RR per i processi time-sharing. Eventualmente è possibile implementare una tecnica di aging per evitare la starvation dei processi time-sharing.

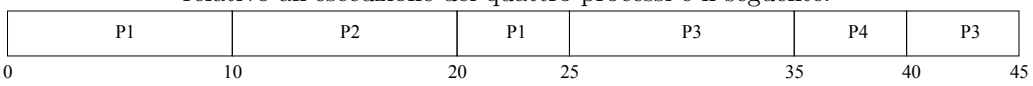
- In coda ready arrivano i processi P_1, P_2, P_3, P_4, con CPU burst e istanti di arrivo specificati in tabella:

	arrivo	burst
P_1	0	15ms
P_2	5	10ms
P_3	15	15ms
P_4	20	5ms

  1. Se i processi terminano nell'ordine P_2, P_1, P_4, P_3, quale può essere l'algoritmo di scheduling? (Si trascuri il tempo di latenza del kernel e, nel caso di processi che arrivano in coda ready nello stesso istante, si dia priorità maggiore al processo con indice inferiore.)
     (a) SRTF
     (b) Scheduling non-preemptive
     (c) RR q=5ms
     (d) RR q=10ms
     (e) Nessuno dei precedenti
  2. (*) Si calcoli il tempo di attesa medio per i processi P_1, P_2, P_3, P_4 della tabella sopra, nel caso di algoritmo RR q=10ms.

  Risposta:

  1. c), d).
  2. Considerando un algoritmo di scheduling RR q=10ms e trascurando il tempo di latenza del kernel, abbiamo cheil diagramma di GANTT relativo all'esecuzione dei quattro processi è il seguente:



  e quindi il tempo di attesa medio è $\frac{10+5+15+15}{4} = \frac{45}{4} = 11,25ms$.

- Si consideri la seguente situazione, dove P_0, P_1, P_2 sono tre 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	A	B	C
P_0	1	2	0	1	5	1
P_1	0	1	0	2	5	2
P_2	2	3	0	2	4	2

Available		(A)
A	B	C
1	5	x

1. Calcolare la matrice R delle richieste.
2. Determinare il minimo valore di x tale che il sistema si trovi in uno stato sicuro.

Risposta:

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

	R	
A	B	C
0	3	1
2	4	2
0	1	2

2. Se $x = 0$, allora non esiste nessuna riga R_i tale che $R_i \leq A$; quindi il sistema si trova in uno stato di deadlock. Se $x = 1$, allora l'unica riga di R minore o uguale a A è la prima. Quindi possiamo eseguire P_0 che, una volta terminato, restituisce le risorse ad esso allocate aggiornando A al valore $(2, 7, 1)$. A questo punto non esiste alcuna riga di R minore o uguale al vettore A e quindi il sistema è in stato di deadlock.

   Il valore minimo di x per cui lo stato risulta sicuro è 2; infatti in questo caso esiste la sequenza sicura $\langle P_0, P_1, P_2 \rangle$. Dapprima si esegue P_0, generando il valore $(2, 7, 2)$ di A, poi si esegue P_1 portando A al valore $(2, 8, 2)$. A questo punto si conclude la sequenza eseguendo P_2 e generando il valore finale di A, ovvero, $(4, 11, 2)$.

- Illustrare brevemente le seguenti strutture dati utilizzate per la gestione dello spazio libero su disco, evidenziandone pregi e difetti:
  - bitmap o vettore di bit,
  - lista concatenata (linked list).

Risposta: Nella bitmap ogni singolo bit rappresenta un blocco del disco: se il blocco è allocato ad un file il bit è 0, se il blocco è libero il bit è 1. Il vantaggio di questa codifica è essenzialmente l'efficienza nel trovare il primo blocco libero dato che la maggior parte delle CPU più diffuse mettono a disposizione delle istruzioni macchina che forniscono l'offset del

primo bit a 1 in una parola. In questo modo il numero del primo blocco libero può essere calcolato come segue:

(numero di bit in una parola)×(numero delle parole con tutti i bit 0)+offset del primo bit a 1

Lo svantaggio è che la bitmap deve essere tenuta in memoria per un utilizzo veloce e quindi può portare ad uno spreco di quest'ultima se il disco è di dimensioni ragguardevoli.

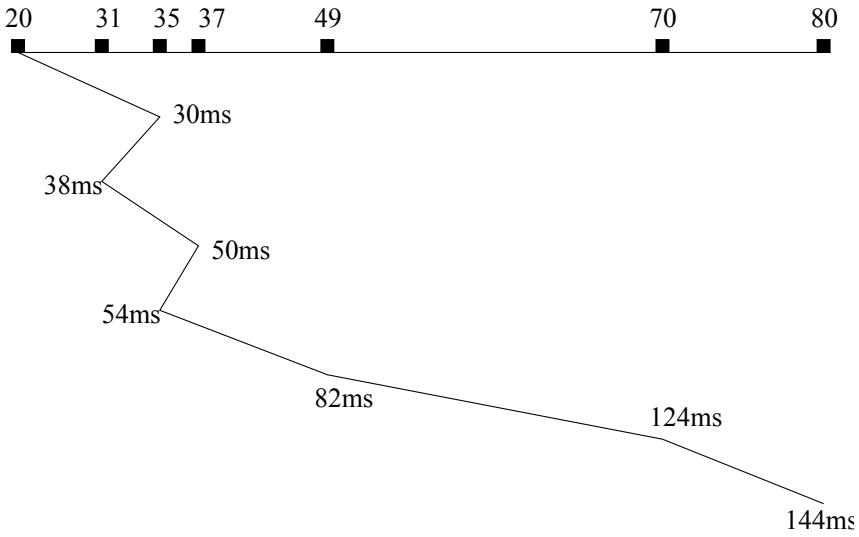
Una soluzione che non richiede un impiego considerevole di memoria è mantenere una lista concatenata dei blocchi del disco liberi (in cui ogni blocco contiene un puntatore al blocco libero successivo). Così facendo è sufficiente mantenere in memoria il puntatore al primo blocco della lista per essere in grado di reperire un blocco libero all'occorrenza. Lo svantaggio è che se si rende necessario attraversare la lista occorre leggere ogni singolo blocco, degradando le prestazioni del sistema in modo considerevole.

- Cosa distingue un sistema debolmente accoppiato da uno strettamente accoppiato? Si faccia qualche esempio di sistema delle due tipologie.

  Risposta: I sistemi strettamente accoppiati sono caratterizzati dalla condivisione di clock e/o memoria (es.: sistemi UMA, NUMA) oppure dalla presenza di linee di comunicazione molto veloci fra i vari nodi del sistema che solitamente risiedono nello stesso rack o stanza e che sono amministrati da una singola organizzazione (es.: multicomputer, cluster of workstations). Viceversa i sistemi debolmente accoppiati sono costituiti da sistemi completi (in cui ogni nodo è sostanzialmente una macchina completa) sparsi in regioni geograficamente anche molto distanti fra loro, controllati da diverse organizzazioni e connessi da reti con linee di comunicazione molto più lente (es.: Internet).

- (*) Si consideri un disco gestito con politica SSTF. Inizialmente la testina è posizionata sul cilindro 20, ascendente; lo spostamento ad una traccia adiacente richiede 2 ms. Al driver di tale disco arrivano richieste per i cilindri 70, 31, 49, 35, 80, rispettivamente agli istanti 0 ms, 30ms, 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 31, 35, 49, 70, 80 come illustrato dal seguente diagramma:



Il punteggio attribuito ai quesiti è il seguente: 3, 3, 3, 3, 2, 2, 4, 4, 6 (totale: 30).