Compito di Sistemi Operativi, Prof. Mancini

5 giugno 2001

1. (punti: -1,4)
   Nei dischi magnetici a teste mobili:

   (a) rispetto alla richiesta di lettura o di scrittura, il trasferimento dei dati inizia con un ritardo chiamato "tempo di ricerca del cilindro";

   (b) rispetto alla richiesta di lettura o di scrittura, il trasferimento dei dati inizia con un ritardo chiamato "tempo di latenza";

   (c) rispetto alla richiesta di lettura o di scrittura, il trasferimento dei dati inizia con un ritardo chiamato "ritardo rotazionale";

   (d) il contributo medio del "tempo di ricerca del cilindro" al tempo di accesso dei singoli caratteri decresce con la dimensione del blocco;

   (e) il contributo medio del "tempo di ricerca del cilindro" al tempo di accesso dei singoli caratteri cresce con la dimensione del blocco;

   (f) il contributo medio del "tempo di latenza" al tempo di accesso dei singoli caratteri decresce con la dimensione del blocco;

   (g) il contributo medio del "tempo di latenza" al tempo di accesso dei singoli caratteri cresce con la dimensione del blocco;

   (h) il contributo medio del "ritardo rotazionale" al tempo di accesso dei singoli caratteri decresce con la dimensione del blocco;

   (i) il contributo medio del "ritardo rotazionale" al tempo di accesso dei singoli caratteri cresce con la dimensione del blocco;

   (j) nessuna delle affermazioni precedenti è corretta.

2. (punti: -1,4)
   Il meccanismo di traduzione degli indirizzi nella gestione della memoria virtuale con paginazione:

   (a) realizza anche la protezione della memoria;

   (b) necessita di meccanismi aggiuntivi per realizzare la protezione della memoria;

   (c) è basato sull'utilizzo di una memoria associativa, che realizza la ricerca del descrittore di pagina;

   (d) è basato sull'utilizzo di una memoria associativa, che è una memoria veloce che mantiene la tabella di descrittori di pagina del processo in esecuzione;

   (e) le informazioni contenute nella memoria associativa includono: l'indice del blocco in memoria principale nel quale la pagina è caricata;

   (f) le informazioni contenute nella memoria associativa includono: lo stato della pagina caricata;

   (g) le informazioni contenute nella memoria associativa includono: l'indice della pagina caricata;

   (h) utilizza una tabella di descrittori di pagina per ogni processo, i cui elementi sono in corrispondenza con gli accessi in memoria del processo;

   (i) utilizza una tabella di descrittori di pagina unica per tutti i processi, i cui elementi sono in corrispondenza con le pagine dei processi;

   (j) nessuna delle affermazioni precedenti è corretta.

3. (punti: -1,4)
   La gestione del processore in ordine di arrivo (FIFO):

(a) non considera il processore una risorsa prerilasciabile ed assegna a tutti i processi pronti uguale priorità;

(b) considera il processore una risorsa prerilasciabile ed assegna a tutti i processi pronti uguale priorità;

(c) considera il processore una risorsa prerilasciabile e non assegna a tutti i processi pronti uguale priorità;

(d) non considera il processore una risorsa prerilasciabile e non assegna a tutti i processi pronti uguale priorità;

(e) è particolarmene adatta per i processi interattivi;

(f) non è particolarmene adatta per i processi interattivi;

(g) penalizza i processi che utilizzano il processore per breve tempo;

(h) non penalizza i processi che utilizzano il processore per breve tempo;

(i) nessuna delle affermazioni precedenti è corretta.

4. (punti: -1,4)
Quali delle seguenti strutture dati appartengono al sistema di archiviazione di Linux:

(a) La *TabellaDeiDescrittoriInMS* contiene, fra l'altro, i puntatori alla posizione corrente degli archivi aperti;

(b) La *TabellaDeiDescrittoriInMS* registra lo stato di assegnazione su disco dei descrittori di archivio;

(c) La *TabellaDeiDescrittoriInMS* contiene le copie dei descrittori di archivio in uso da parte di almeno un processo;

(d) La *MappaDeiDescrittoriDiArchivio* registra lo stato di assegnazione su disco dei descrittori di archivio;

(e) La *MappaDeiDescrittoriDiArchivio* contiene le copie dei descrittori di archivio in uso da parte di almeno un processo;

(f) La *MappaDeiDescrittoriDiArchivio* contiene, fra l'altro, i puntatori alla posizione corrente degli archivi aperti;

(g) La *TabellaDeiCollegamenti* contiene le copie dei descrittori di archivio in uso da parte di almeno un processo;

(h) La *TabellaDeiCollegamenti* registra lo stato di assegnazione su disco dei descrittori di archivio;

(i) La *TabellaDeiCollegamenti* contiene, fra l'altro, i puntatori alla posizione corrente degli archivi aperti;

(j) nessuna delle affermazioni precedenti è corretta.

5. (punti: -1,4)
Ad ogni processo del sistema operativo Linux viene associata una quadrupla (*proprietario reale, gruppo reale, proprietario effettivo, gruppo effettivo*):

(a) il proprietario reale e il gruppo reale contengono informazioni (ad esempio, username) relative all'utente reale, mentre il proprietario effettivo e il gruppo effettivo contengono informazioni (UID e GID) utilizzate internamente dal sistema per identificare in modo univoco l'utente;

(b) il proprietario reale e il gruppo reale contengono informazioni (UID e GID) utilizzate internamente dal sistema per identificare in modo univoco l'utente, mentre il proprietario effettivo e il gruppo effettivo contengono informazioni (ad esempio, username) relative all'utente reale;

(c) l'identificazione degli utenti si propaga ai processi, nell'istante di creazione ogni processo è attribuito ad un utente proprietario assegnando opportunamente i valori alla quadrupla che non può mai essere modificata durante l'esecuzione del processo;

(d) l'identificazione degli utenti si propaga ai processi, nell'istante di creazione ogni processo è attribuito ad un utente proprietario assegnando opportunamente i valori alla quadrupla che può essere modificata durante l'esecuzione del processo;

(e) i meccanismi di protezione di Linux non consentono la migrazione dei processi in domini di protezione diversi da quelli dei loro proprietari;

(f) si assuma che un processo invochi la chiamata di sistema **exec** per eseguire il programma contenuto in un archivio F_i; **exec** assegna al proprietario reale, e al gruppo reale i valori corrispondenti al proprietario dell'archivio F_i se e solo se è vera la condizione SETUID contenuta nell' i-node di F_i;

(g) si assuma che un processo invochi la chiamata di sistema **exec** per eseguire il programma contenuto in un archivio F_i; **exec** assegna al proprietario effettivo, e al gruppo effettivo i valori corrispondenti al proprietario dell'archivio F_i se e solo se è vera la condizione SETUID contenuta nell' i-node di F_i;

(h) nessuna delle affermazioni precedenti è corretta.

6. (punti: 6)
Usando un massimo di 120 parole, confrontare i meccanismi di interazione tra processi basati su scambio di messaggi con quelli basati su condivisione della memoria, evidenziando eventuali vantaggi e svantaggi degli uni rispetto agli altri. Fornire esempi di entrambi questi paradigmi di interazione.

7. (punti: 6)
Illustrare in al più 70 parole i concetti di indirizzo fisico, indirizzo logico ed indirizzo effettivo di memoria.

8. (punti: 6)
Considerare un insieme di cinque processi P_1, P_2, P_3, P_4, P_5 con i seguenti tempi di arrivo e tempi di esecuzione in millisecondi:

Processo	Tempo di arrivo	Tempo di esecuzione
P_1	4	20
P_2	17	7
P_3	11	14
P_4	5	7
P_5	23	13

Assegnare questo insieme di processi ad un processore in base alla politica Round Robin considerando un quanto di tempo di 6 millisecondi.

Calcolare il valor medio del tempo di attesa ed il valor medio del tempo di turnaround dei processi.

9. Considerare la seguente stringa di riferimenti alla memoria di un processo in un sistema con memoria virtuale S = 4 12 3 2 1 5 3 4 9 1 2 5 9 5 1 2 10 12 3 9

(a) (punti: 3)
Illustrare il comportamento dell'algoritmo LRU di sostituzione delle pagine per una memoria fisica di 5 blocchi. Calcolare il numero di page fault che si verificano.

(b) (punti: 3)
Illustrare il comportamento dell'algoritmo Second Chance di sostituzione delle pagine per una memoria fisica di 5 blocchi. Calcolare il numero di page fault che si verificano.

10. (punti: 16)
Il seguente programma Java implementa il meccanismo delle pipe di uno shell Unix. Una pipe tra un comando c1 ed un comando c2, scritta c1|c2, realizza un comando di comunicazione asincrona attraverso il quale l'output di c1 viene passato in input a c2. Scrivere i metodi *scrivi* e *leggi* della classe Coda in modo da implementare un tale canale. Si assume per semplicita' che l'input e l'output dei comandi sia una sequenza di al piu' 10 interi.

```
class TestPipe {
    public static void main (String [] args) {
 Comando c1 = new Comando();
 Comando c2 = new Comando();
 (new ComandoPipe(c1, c2)).start();
    }
}
class Coda {
    static Coda STANDARD_INPUT = new Coda();
    static Coda STANDARD_OUTPUT = new Coda();

    private int[] A = new int[10];
    private int in = 0;
    private int out = -1;

    public synchronized void scrivi (int v) {
        // ???
        A[in++] = v;
        // ???
    }
    public synchronized int leggi () {
        // ???
        return A[++out];
    }
}

class Comando extends Thread {
    public Coda input;
    public Coda output;

    public Comando () {
        input = Coda.STANDARD_INPUT;
        output = Coda.STANDARD_OUTPUT;
    }
}

class ComandoPipe extends Comando {
    Comando first, second;

    public ComandoPipe (Comando c1, Comando c2) {
        first = c1;
        second = c2;
        second.input = first.output = new Coda();
    }

    public void run () {
        first.start();
        second.start();
    }
}
```

Cognome: **Nome:** **Matricola:**

Risposte Compito di Sistemi Operativi, 5/6/2001

1. (punti: -1,4) **a** **b** **c** **d** **e** **f** **g** **h** i **j** **k**
2. (punti: -1,4) **a** **b** **c** **d** **e** **f** **g** **h** i **j** **k**
3. (punti: -1,4) **a** **b** **c** **d** **e** **f** **g** **h** i **j** **k**
4. (punti: -1,4) **a** **b** **c** **d** **e** **f** **g** **h** i **j** **k**
5. (punti: -1,4) **a** **b** **c** **d** **e** **f** **g** **h** i **j** **k**
6. (punti: 6)

7. (punti: 6)

8. (punti: 6)

9. (a) (punti: 3)

   (b) (punti: 3)