

Tema d'esame
di Sistemi Operativi I e II e Laboratorio
12 luglio 2004
Docenti Matteo Vaccari e Marco Benini

Nome e Cognome:

Matricola:

Prima parte

1. (4 punti)

Si consideri il seguente programma:

```
#define N 10000
int main() {
    int i, j;
    char m[N][N];
    for (i=0; i<N; i++)
        for (j=0; j<N; j++)
            m[i][j] = 0; // **
}
```

Su una macchina che ha una memoria RAM di 90MB, il programma ha una performance che peggiora di molto se sostituisco la riga marcata “**” con la seguente:

```
m[j][i] = 0;
```

Spiegare perché

2. (4 punti)

Si descriva brevemente il meccanismo di "buffer cache" di Unix:

(a) a che cosa serve?

(b) come funziona?

(c) quando eseguo una `write(2)` per scrivere su un file, quando viene modificato fisicamente il file sul disco? disco

(d) cosa posso fare per assicurarmi, dopo una `write(2)`, che il file sia stato effettivamente modificato?

3. (4 punti)

(a) Che cosa fa la chiamata di sistema `read(2)` di Unix

(b) che argomenti richiede, e qual'è il loro significato

(c) che valore restituisce? come viene segnalato un errore?

(d) che differenze ci sono fra la chiamata `read(2)` e la `fprintf(3)` ?

4. (4 punti)

Si scriva un programma C che

- genera un processo figlio,
- controlla l'input e l'output del processo figlio con due pipe.
- Il processo figlio esegue il comando `bc(1)`
- il processo genitore manda al figlio le righe `"3 * 3\n"`, `"7 * 7\n"` e stampa in output il risultato ottenuto dal figlio

5. (4 punti)

Un museo deve limitare l'accesso a non più di 100 visitatori in contemporanea. Per questo motivo, quando un visitatore si presenta all'ingresso, gli viene dato un biglietto con un numero compreso fra 0 e 99.

Se tutti i 100 biglietti sono già stati dati, il visitatore aspetta che ne ritorni disponibile uno.

Quando un visitatore esce dal museo, restituisce il suo biglietto all'ingresso.

Scrivere due procedure *ingresso()* e *uscita()* che implementano questo meccanismo per mezzo di semafori. E' da evitarsi il busy wait.

