Prova di Sistemi Operativi del 22 Marzo 2005, TRACCIA A

Un campo da golf possiede un pool di N palline da golf, che possono essere assegnate a un certo numero di giocatori. I singoli giocatori possono affittare un certo numero *x* di palline per poter giocare. Dopo aver giocato, i giocatori restituiscono le palline che vengono quindi messe a disposizione di eventuali giocatori in attesa. Un giocatore non può ritirare le palline richieste se queste non sono disponibili nel numero voluto. Ad esempio se in un certo momento sono disponibili 5 palline, la richiesta di 6 palline da parte di un giocatore non potrà essere esaudita, mentre un giocatore che faccia richiesta di 3 palline sarà servito normalmente.

Si progettino le seguenti classi:

-la classe *Pool* che contenga almeno i due metodi

```
void noleggiaPalline(int numPalline)
void restituisciPalline(int numPalline)
```

Il primo metodo consente di noleggiare un certo numero di palline. Se la quantità di palline richiesta non è disponibile il giocatore viene messo in attesa. Il secondo metodo prevede la restituzione del numero di palline corrispondente.

-la classe *Giocatore* che simuli il comportamento casuale di un giocatore attraverso un thread.

Bonus: si individuino le possibili cause di starvation per i giocatori e si fornisca una soluzione dove tali problemi sono eliminati.

Descrivere in maniera dettagliata il meccanismo di gestione della memoria denominato paginazione e l'architettura di base per l'implementazione della paginazione tramite la tabella delle pagine. Discutere il comportamento della paginazione rispetto alle problematiche di frammentazione della memoria.

Prova di Sistemi Operativi del 22 Marzo 2005, TRACCIA B

Una sala da bowling possiede 2 piste e N palle da bowling. Ogni pista può essere assegnata a una certa squadra, ammesso che sia disponibile un numero di palle da bowling pari al numero di giocatori per la data squadra. Ad esempio, se le palle a disposizione sono 10, e in un certo momento una squadra di 6 giocatori occupa la pista n.1, una eventuale squadra di 5 giocatori non potrà usufruire della pista n.2, pur essendo questa libera.

Si progettino le seguenti classi:

-la classe *Sala* che contenga almeno i due metodi

```
int richiediPista(int numGiocatori)
void liberaPista(int codicePista, int numGiocatori)
```

Il primo metodo consente di richiedere una pista con un certo numero di giocatori. Se la quantità di palle richiesta non è disponibile, oppure non ci sono piste libere, la squadra viene messa in attesa. Il secondo metodo prevede la liberazione di una pista occupata e la restituzione delle palle non più utilizzate.

-la classe *Squadra* che simuli il comportamento casuale di una squadra attraverso un thread.

Bonus: si individuino le possibili cause di starvation per le squadre e si fornisca una soluzione dove tali problemi sono eliminati.

Descrivere in maniera dettagliata un semplice algoritmo a turni con quanto di tempo per lo scheduling della CPU. Discutere pregi e difetti di tale tecnica e l'influenza del tempo di context switch sulle sue prestazioni.

Prova pratica di Sistemi Operativi – 22 Marzo 2005 – TRACCIA A

Si scriva in Perl il comando wordCount, il quale ha la seguente sintassi:

```
wordCount [opzione] nomeFile [parola]
```

devono essere previste le seguenti modalità di funzionamento:

Nessuna opzione presente: stampa in output il numero di parole presenti in *nomeFile*.
Opzione –w: stampa in output il numero di occorrenze della parola *parola*.

Prova pratica di Sistemi Operativi – 22 Marzo 2005 – TRACCIA B

Si scriva in Perl il comando wordCount, il quale ha la seguente sintassi:

```
wordCount [opzione] nomeFile
```

devono essere previste le seguenti modalità di funzionamento:

Nessuna opzione presente: stampa in output il numero di parole presenti in *nomeFile*.
Opzione –l: stampa in output la linea più lunga e la sua lunghezza.