

SISTEMI OPERATIVI IDT/IEL - 1^a prova intermedia, 13 novembre 2007 A1

Esercizio 1

Sia dato un sistema nel quale a ciascun processo è assegnato, in maniera statica, uno di due possibili livelli di priorità ("ALTA" e "BASSA"). La priorità tra i due livelli è gestita con prerilascio. Per ogni livello di priorità è prevista inoltre una specifica politica di scheduling. In particolare, la coda dei processi pronti ad alta priorità è gestita in modalità Round Robin con quanto di tempo $q=1.5$, mentre quella dei processi pronti a bassa priorità è gestita secondo uno schema FCFS.

Siano inoltre dati 4 processi P1, P2, P3 e P4, con i tempi di arrivo (T_A), i tempi di elaborazione richiesti per il loro completamento (T_{CPU}) e le priorità riportati nella tabella sottostante:

Processo	T_A	T_{CPU}	Priorità
P1	0	5	BASSA
P2	1	5	BASSA
P3	3	6	ALTA
P4	4	3	ALTA

Si determinino quindi

- la traccia di esecuzione dei processi;
- il tempo di completamento (*turnaround time*) per ciascun processo.

Esercizio 2

Si definisca una classe Java che, dato un vettore V di interi di dimensione N, in un flusso di esecuzione dedicato, costruisca un nuovo vettore W di dimensione N+1 tale che:

$$W[0] = V[0]$$

$$W[i] = V[i-1] + V[i] \quad \text{per } 0 < i < N$$

$$W[N] = V[N-1]$$

e quindi stampi il contenuto del vettore W.

Si realizzi poi un programma Java che, partendo da un vettore di un solo elemento inizializzato ad 1, iteri la procedura codificata nella classe di cui sopra fino ad ottenere un vettore di dimensione K (ovvero, stampi il triangolo di Tartaglia fino alla potenza K-esima).

SISTEMI OPERATIVI IDT/IEL - 1^a prova intermedia, 13 novembre 2007 A3

Esercizio 1

Sia dato un sistema nel quale a ciascun processo è assegnato, in maniera statica, uno di due possibili livelli di priorità ("ALTA" e "BASSA"). La priorità tra i due livelli è gestita con prerilascio. Per ogni livello di priorità è prevista inoltre una specifica politica di scheduling. In particolare, la coda dei processi pronti ad alta priorità è gestita in modalità Round Robin con quanto di tempo $q=1.5$, mentre quella dei processi pronti a bassa priorità è gestita secondo uno schema SJF senza prerilascio (o *preemption*).

Siano inoltre dati 4 processi P1, P2, P3 e P4, con i tempi di arrivo (T_A), i tempi di elaborazione richiesti per il loro completamento (T_{CPU}) e le priorità riportati nella tabella sottostante:

Processo	T_A	T_{CPU}	Priorità
P1	0	5	BASSA
P2	1	5	BASSA
P3	3	6	ALTA
P4	4	3	ALTA

Si determinino quindi

- la traccia di esecuzione dei processi;
- il tempo di completamento (*turnaround time*) per ciascun processo.

Esercizio 2

Si definisca una classe Java che, dato un vettore V di interi di dimensione N, in un flusso di esecuzione dedicato, costruisca un nuovo vettore W di dimensione N+1 tale che:

$$W[0] = V[0]$$

$$W[i] = V[i-1] + V[i] \quad \text{per } 0 < i < N$$

$$W[N] = V[N-1]$$

e quindi stampi il contenuto del vettore W.

Si realizzi poi un programma Java che, partendo da un vettore di un solo elemento inizializzato ad 1, iteri la procedura codificata nella classe di cui sopra fino ad ottenere un vettore di dimensione K (ovvero, stampi il triangolo di Tartaglia fino alla potenza K-esima).

SISTEMI OPERATIVI IDT/IEL - 1^a prova intermedia, 13 novembre 2007 cs

Esercizio 1

Sia dato un sistema nel quale a ciascun processo è assegnato, in maniera statica, uno di due possibili livelli di priorità (“ALTA” e “BASSA”). La priorità tra i due livelli è gestita con prerilascio. Per ogni livello di priorità è prevista inoltre una specifica politica di scheduling. In particolare, la coda dei processi pronti ad alta priorità è gestita in modalità Round Robin con quanto di tempo $q=1.5$, mentre quella dei processi pronti a bassa priorità è gestita secondo uno schema FCFS.

Siano inoltre dati 4 processi P1, P2, P3 e P4, con i tempi di arrivo (T_A), i tempi di elaborazione richiesti per il loro completamento (T_{CPU}) e le priorità riportati nella tabella sottostante:

Processo	T_A	T_{CPU}	Priorità
P1	0	5	BASSA
P2	1	5	BASSA
P3	3	6	ALTA
P4	4	3	ALTA

Si determinino quindi

- la traccia di esecuzione dei processi;
- il tempo di attesa (*waiting time*) per ciascun processo.

Esercizio 2

Si definisca una classe Java che, dato un vettore V di interi di dimensione N, in un flusso di esecuzione dedicato, costruisca un nuovo vettore W di dimensione N+1 tale che:

$$W[0] = V[0]$$

$$W[i] = V[i-1] + V[i] \quad \text{per } 0 < i < N$$

$$W[N] = V[N-1]$$

e quindi stampi il contenuto del vettore W.

Si realizzi poi un programma Java che, partendo da un vettore di un solo elemento inizializzato ad 1, iteri la procedura codificata nella classe di cui sopra fino ad ottenere un vettore di dimensione K (ovvero, stampi il triangolo di Tartaglia fino alla potenza K-esima).

SISTEMI OPERATIVI IDT/IEL - 1^a prova intermedia, 13 novembre 2007 D7

Esercizio 1

Sia dato un sistema nel quale a ciascun processo è assegnato, in maniera statica, uno di due possibili livelli di priorità ("ALTA" e "BASSA"). La priorità tra i due livelli è gestita con prerilascio. Per ogni livello di priorità è prevista inoltre una specifica politica di scheduling. In particolare, la coda dei processi pronti ad alta priorità è gestita in modalità Round Robin con quanto di tempo $q=1.5$, mentre quella dei processi pronti a bassa priorità è gestita secondo uno schema SJF senza prerilascio (o *preemption*).

Siano inoltre dati 4 processi P1, P2, P3 e P4, con i tempi di arrivo (T_A), i tempi di elaborazione richiesti per il loro completamento (T_{CPU}) e le priorità riportati nella tabella sottostante:

Processo	T_A	T_{CPU}	Priorità
P1	0	5	BASSA
P2	1	5	BASSA
P3	3	6	ALTA
P4	4	3	ALTA

Si determinino quindi

- la traccia di esecuzione dei processi;
- il tempo di attesa (*waiting time*) per ciascun processo.

Esercizio 2

Si definisca una classe Java che, dato un vettore V di interi di dimensione N, in un flusso di esecuzione dedicato, costruisca un nuovo vettore W di dimensione N+1 tale che:

$$W[0] = V[0]$$

$$W[i] = V[i-1] + V[i] \quad \text{per } 0 < i < N$$

$$W[N] = V[N-1]$$

e quindi stampi il contenuto del vettore W.

Si realizzi poi un programma Java che, partendo da un vettore di un solo elemento inizializzato ad 1, iteri la procedura codificata nella classe di cui sopra fino ad ottenere un vettore di dimensione K (ovvero, stampi il triangolo di Tartaglia fino alla potenza K-esima).