

## SISTEMI OPERATIVI IDT/IEL - Prima prova intermedia, 1 giugno 2007 As

### 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  $q=1$ , mentre quella dei processi pronti a bassa priorità è gestita secondo uno schema SJF con prerilascio.

Supponendo di avere 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, si determini la traccia di esecuzione dei processi.

<b>Processo</b>	<b><math>T_A</math></b>	<b><math>T_{CPU}</math></b>	<b>Priorità</b>
P1	0	5	BASSA
P2	1	2	BASSA
P3	2	3	ALTA
P4	3,5	4	ALTA

### Esercizio 2

Si definisca classe Java che, in un flusso di esecuzione dedicato, effettui la somma di due numeri interi

Si realizzi quindi un programma Java che, dato un vettore di interi di dimensione N, ne inizializzi anzitutto gli elementi con numeri casuali compresi tra 1 e 10, quindi calcoli la somma degli elementi del vettore, e infine mostri sullo schermo il risultato dell'elaborazione. Per l'operazione algebrica di somma il programma potrà utilizzare esclusivamente la classe di cui sopra, sfruttando la proprietà associativa della somma.

Per la risoluzione del problema non è consentito l'uso della ricorsione.

## SISTEMI OPERATIVI IDT/IEL - Prima prova intermedia, 1 giugno 2007 BP

### 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  $q=2$ , mentre quella dei processi pronti a bassa priorità è gestita secondo uno schema SJF con prerilascio.

Supponendo di avere 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, si determini la traccia di esecuzione dei processi.

<b>Processo</b>	<b><math>T_A</math></b>	<b><math>T_{CPU}</math></b>	<b>Priorità</b>
P1	0	5	BASSA
P2	1	2	BASSA
P3	2	3	ALTA
P4	3,5	4	ALTA

### Esercizio 2

Si definisca classe Java che, in un flusso di esecuzione dedicato, effettui il prodotto di due numeri interi

Si realizzi quindi un programma Java che, dato un vettore di interi di dimensione N, ne inizializzi anzitutto gli elementi con numeri casuali compresi tra 1 e 10, quindi calcoli il prodotto degli elementi del vettore, e infine mostri sullo schermo il risultato dell'elaborazione. Per l'operazione algebrica di prodotto il programma potrà utilizzare esclusivamente la classe di cui sopra, sfruttando la proprietà associativa del prodotto.

Per la risoluzione del problema non è consentito l'uso della ricorsione.

## SISTEMI OPERATIVI IDT/IEL - Prima prova intermedia, 1 giugno 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  $q=1$ , mentre quella dei processi pronti a bassa priorità è gestita secondo uno schema SJF senza prerilascio.

Supponendo di avere 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, si determini la traccia di esecuzione dei processi.

<b>Processo</b>	<b><math>T_A</math></b>	<b><math>T_{CPU}</math></b>	<b>Priorità</b>
P1	0	5	BASSA
P2	1	2	BASSA
P3	2	3	ALTA
P4	3,5	4	ALTA

### Esercizio 2

Si definisca classe Java che, in un flusso di esecuzione dedicato, effettui la somma di due numeri interi

Si realizzi quindi un programma Java che, dato un vettore di interi di dimensione N, ne inizializzi anzitutto gli elementi con numeri casuali compresi tra 1 e 10, quindi calcoli la somma degli elementi del vettore, e infine mostri sullo schermo il risultato dell'elaborazione. Per l'operazione algebrica di somma il programma potrà utilizzare esclusivamente la classe di cui sopra, sfruttando la proprietà associativa della somma.

Per la risoluzione del problema non è consentito l'uso della ricorsione.

## SISTEMI OPERATIVI IDT/IEL - Prima prova intermedia, 1 giugno 2007 DP

### 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  $q=2$ , mentre quella dei processi pronti a bassa priorità è gestita secondo uno schema SJF senza prerilascio.

Supponendo di avere 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, si determini la traccia di esecuzione dei processi.

<b>Processo</b>	<b><math>T_A</math></b>	<b><math>T_{CPU}</math></b>	<b>Priorità</b>
P1	0	5	BASSA
P2	1	2	BASSA
P3	2	3	ALTA
P4	3,5	4	ALTA

### Esercizio 2

Si definisca classe Java che, in un flusso di esecuzione dedicato, effettui il prodotto di due numeri interi

Si realizzi quindi un programma Java che, dato un vettore di interi di dimensione N, ne inizializzi anzitutto gli elementi con numeri casuali compresi tra 1 e 10, quindi calcoli il prodotto degli elementi del vettore, e infine mostri sullo schermo il risultato dell'elaborazione. Per l'operazione algebrica di prodotto il programma potrà utilizzare esclusivamente la classe di cui sopra, sfruttando la proprietà associativa del prodotto.

Per la risoluzione del problema non è consentito l'uso della ricorsione.

## SISTEMI OPERATIVI IDT/IEL - Prima prova intermedia, 1 giugno 2007 AP

### 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  $q=1$ , mentre quella dei processi pronti a bassa priorità è gestita secondo uno schema SJF con prerilascio.

Supponendo di avere 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, si determini la traccia di esecuzione dei processi.

<b>Processo</b>	<b><math>T_A</math></b>	<b><math>T_{CPU}</math></b>	<b>Priorità</b>
P1	0	5	BASSA
P2	1	2	BASSA
P3	2	3	ALTA
P4	3,5	4	ALTA

### Esercizio 2

Si definisca classe Java che, in un flusso di esecuzione dedicato, effettui il prodotto di due numeri interi

Si realizzi quindi un programma Java che, dato un vettore di interi di dimensione N, ne inizializzi anzitutto gli elementi con numeri casuali compresi tra 1 e 10, quindi calcoli il prodotto degli elementi del vettore, e infine mostri sullo schermo il risultato dell'elaborazione. Per l'operazione algebrica di prodotto il programma potrà utilizzare esclusivamente la classe di cui sopra, sfruttando la proprietà associativa del prodotto.

Per la risoluzione del problema non è consentito l'uso della ricorsione.

## SISTEMI OPERATIVI IDT/IEL - Prima prova intermedia, 1 giugno 2007 vs

### 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  $q=2$ , mentre quella dei processi pronti a bassa priorità è gestita secondo uno schema SJF con prerilascio.

Supponendo di avere 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, si determini la traccia di esecuzione dei processi.

<b>Processo</b>	<b><math>T_A</math></b>	<b><math>T_{CPU}</math></b>	<b>Priorità</b>
P1	0	5	BASSA
P2	1	2	BASSA
P3	2	3	ALTA
P4	3,5	4	ALTA

### Esercizio 2

Si definisca classe Java che, in un flusso di esecuzione dedicato, effettui la somma di due numeri interi

Si realizzi quindi un programma Java che, dato un vettore di interi di dimensione N, ne inizializzi anzitutto gli elementi con numeri casuali compresi tra 1 e 10, quindi calcoli la somma degli elementi del vettore, e infine mostri sullo schermo il risultato dell'elaborazione. Per l'operazione algebrica di somma il programma potrà utilizzare esclusivamente la classe di cui sopra, sfruttando la proprietà associativa della somma.

Per la risoluzione del problema non è consentito l'uso della ricorsione.

## SISTEMI OPERATIVI IDT/IEL - Prima prova intermedia, 1 giugno 2007 CP

### 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  $q=1$ , mentre quella dei processi pronti a bassa priorità è gestita secondo uno schema SJF senza prerilascio.

Supponendo di avere 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, si determini la traccia di esecuzione dei processi.

<b>Processo</b>	<b><math>T_A</math></b>	<b><math>T_{CPU}</math></b>	<b>Priorità</b>
P1	0	5	BASSA
P2	1	2	BASSA
P3	2	3	ALTA
P4	3,5	4	ALTA

### Esercizio 2

Si definisca classe Java che, in un flusso di esecuzione dedicato, effettui il prodotto di due numeri interi

Si realizzi quindi un programma Java che, dato un vettore di interi di dimensione N, ne inizializzi anzitutto gli elementi con numeri casuali compresi tra 1 e 10, quindi calcoli il prodotto degli elementi del vettore, e infine mostri sullo schermo il risultato dell'elaborazione. Per l'operazione algebrica di prodotto il programma potrà utilizzare esclusivamente la classe di cui sopra, sfruttando la proprietà associativa del prodotto.

Per la risoluzione del problema non è consentito l'uso della ricorsione.

## SISTEMI OPERATIVI IDT/IEL - Prima prova intermedia, 1 giugno 2007 ds

### 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  $q=2$ , mentre quella dei processi pronti a bassa priorità è gestita secondo uno schema SJF senza prerilascio.

Supponendo di avere 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, si determini la traccia di esecuzione dei processi.

<b>Processo</b>	<b><math>T_A</math></b>	<b><math>T_{CPU}</math></b>	<b>Priorità</b>
P1	0	5	BASSA
P2	1	2	BASSA
P3	2	3	ALTA
P4	3,5	4	ALTA

### Esercizio 2

Si definisca classe Java che, in un flusso di esecuzione dedicato, effettui la somma di due numeri interi

Si realizzi quindi un programma Java che, dato un vettore di interi di dimensione N, ne inizializzi anzitutto gli elementi con numeri casuali compresi tra 1 e 10, quindi calcoli la somma degli elementi del vettore, e infine mostri sullo schermo il risultato dell'elaborazione. Per l'operazione algebrica di somma il programma potrà utilizzare esclusivamente la classe di cui sopra, sfruttando la proprietà associativa della somma.

Per la risoluzione del problema non è consentito l'uso della ricorsione.