Iniziato	martedì, 14 gennaio 2025, 14:03
Stato	Completato
Terminato	martedì, 14 gennaio 2025, 14:43
Tempo impiegato	39 min. 40 secondi
Valutazione	**12,50** su un massimo di 23,00 (**54,35**%)

Domanda **1**

Risposta non data

Punteggio max.: 2,00

In hard disk grande 512 Gigabyte, per scrivere il numero di un blocco vengono usati 27 bit, arrotondati al minimo numero di byte necessario. L'hard disk adotta una allocazione indicizzata semplice, e di un file A si sa che nel suo blocco indice 20 byte vengono usati per tenere traccia dei blocchi di dati di A. Quanto può essere grande al massimo A?

Scegli un'alternativa:

a. 20 Kilobyte

b. 24 Kilobyte

c. 18 Kilobyte

d. 16 Kilobyte

Risposta errata.

La risposta corretta è: 20 Kilobyte

Domanda **2**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Dopo l'esecuzione dei seguenti comandi in un ambiente Unix (come visti a lezione):

1: cd /tmp
2: mkdir newfolder
3: cd newfolder
4: echo "ciao" > pippo // crea un nuovo file di nome *pippo* contenente la stringa *ciao*
5: ln pippo paperino
6: ln -s /tmp/newfolder folder2
7: cp paperino topolino
8: echo "salve" >> topolino // aggiunge "salve" a fondo file
9: rm pippo
10: cat paperino // *cat* stampa il contenuto del file passato come argomento
11: mkdir ../folder3

Scegli un'alternativa:

a. 1. il link-counter dell'i-node di *paperino* è: 1
2. il link counter di *newfolder* è: 2
3. l'output del comando 10 è: "ciao"
4. il link counter di tmp è: aumentato di 3

b. 1. il link-counter dell'i-node di *paperino* è: 2
2. il link counter di *newfolder* è: 2
3. l'output del comando 10 è: "ciao"
4. il link counter di tmp è: aumentato di 3

c. 1. il link-counter dell'i-node di *paperino* è: 2
2. il link counter di *newfolder* è: 2
3. l'output del comando 10 è: "ciao"
4. il link counter di tmp è: aumentato di 1

d. 1. il link-counter dell'i-node di *paperino* è: 1 ✔
2. il link counter di *newfolder* è: 2
3. l'output del comando 10 è: "ciao"
4. il link counter di tmp è: aumentato di 2

Risposta corretta.

La risposta corretta è:
1. il link-counter dell'i-node di *paperino* è: 1
2. il link counter di *newfolder* è: 2
3. l'output del comando 10 è: "ciao"
4. il link counter di tmp è: aumentato di 2

Domanda **3**

Risposta errata

Punteggio ottenuto -1,00 su 2,00

Le caratteristiche fondamentali dell'algoritmo di scheduling RR sono:

Scegli un'alternativa:

a.
1. al diminuire del quanto di tempo aumenta l'overhead dei context switch ✖
2. fornisce mediamente waiting time migliori di quelli di SJF
3. all'aumentare del quanto di tempo tende a comportarsi come FCFS
4. è l'algoritmo di base naturale dei sistemi time sharing

b.
1. all'aumentare del quanto di tempo tende a comportarsi come FCFS
2. è l'algoritmo di base naturale dei sistemi time sharing
3. al diminuire del quanto di tempo diminuisce l'overhead dei context switch
4. fornisce mediamente un buon tempo di risposta

c.
1. al diminuire del quanto di tempo aumenta l'overhead dei context switch
2. fornisce mediamente un buon tempo di risposta
3. al diminuire del quanto di tempo tende a comportarsi come FCFS
4. è l'algoritmo di base naturale dei sistemi time sharing

d.
1. fornisce mediamente un buon tempo di risposta
2. è l'algoritmo di base naturale dei sistemi time sharing
3. al diminuire del quanto di tempo aumenta l'overhead dei context switch
4. all'aumentare del quanto di tempo tende a comportarsi come FCFS

Risposta errata.

La risposta corretta è:
1. fornisce mediamente un buon tempo di risposta
2. è l'algoritmo di base naturale dei sistemi time sharing
3. al diminuire del quanto di tempo aumenta l'overhead dei context switch
4. all'aumentare del quanto di tempo tende a comportarsi come FCFS

Domanda **4**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Tra le ragioni per cui un sistema operativo può finire in una condizione di thrashing troviamo:

Scegli un'alternativa:

a. un elevato numero di processi attivi rispetto alla dimensione della memoria principale, e l'adozione di un algoritmo di sostituzione delle pagine di tipo FIFO

b. un elevato numero di processi attivi rispetto alla dimensione della memoria principale, e l'adozione di una politica di allocazione globale o locale dei frame ✔

c. un elevato numero di processi I/O bound rispetto alla dimensione della memoria principale, e l'adozione di una politica di allocazione globale o locale dei frame

d. un elevato numero di processi CPU bound rispetto alla dimensione della memoria principale, e l'adozione di una politica di allocazione globale o locale dei frame

Risposta corretta.

La risposta corretta è: un elevato numero di processi attivi rispetto alla dimensione della memoria principale, e l'adozione di una politica di allocazione globale o locale dei frame

Domanda **5**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Dire che un sistema adotta un *binding dinamico degli indirizzi* significa che:

Scegli un'alternativa:

a. i programmi che girano su quel sistema usano solo indirizzi relativi, che vengono tradotti in indirizzi assoluti durante l'esecuzione delle istruzioni che usano quegli indirizzi ✔

b. il sistema operativo traduce gli indirizzi usati dai processi in esecuzione in modo dinamico, ossia solo nel momento in cui tali processi stanno per passare dallo stato "ready to run" allo stato "running"

c. i programmi che girano su quel sistema usano solo la memoria dinamica per allocare le variabili, ottimizzando così lo spazio occupato da ciascun programma

d. gli indirizzi relativi usati dalle istruzioni dei programmi in esecuzione possono cambiare da una esecuzione alla successiva.

Risposta corretta.

La risposta corretta è: i programmi che girano su quel sistema usano solo indirizzi relativi, che vengono tradotti in indirizzi assoluti durante l'esecuzione delle istruzioni che usano quegli indirizzi

Domanda **6**

Parzialmente corretta

Punteggio ottenuto 1,00 su 2,00

Di un sistema è noto che la tabella delle pagine più grande del sistema occupa esattamente 4 frame, il numero di un frame è scritto su 4 byte usando solo i primi 26 bit, e nel sistema sono presenti in media 4 processi che insieme producono una frammentazione interna complessiva media di 2 Kilobyte.

lo spazio logico del sistema è grande: ✖

lo spazio fisico del sistema è grande: 64 Gigabyte ✔

32 Gigabyte	nessuno dei valori proposti
2 Megabyte	non si può ricavare dai dati del problema
16 Gigabyte	
1 Megabyte	4 Megabyte

Risposta parzialmente esatta.

Hai selezionato correttamente 1.
La risposta corretta è:

Di un sistema è noto che la tabella delle pagine più grande del sistema occupa esattamente 4 frame, il numero di un frame è scritto su 4 byte usando solo i primi 26 bit, e nel sistema sono presenti in media 4 processi che insieme producono una frammentazione interna complessiva media di 2 Kilobyte.

lo spazio logico del sistema è grande: [1 Megabyte]
lo spazio fisico del sistema è grande: [64 Gigabyte]

Domanda **7**

Risposta non data

Punteggio max.: 2,00

Su un hard disk che adotta una allocazione concatenata (senza FAT) è memorizzato un file A della dimensione di 0x4000 byte, e si sa che nell'ultimo blocco di A sono presenti 16 byte del file. Si sa inoltre che per scrivere il numero di un blocco vengono usati 27 bit, arrotondati al minimo numero di byte necessario. Quanto è grosso l'hard disk?

Scegli un'alternativa:

a. 128 Gigabyte

b. 1 Terabyte

c. 512 Gigabyte

d. 256 Gigabyte

Risposta errata.

La risposta corretta è: 512 Gigabyte

Domanda **8**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Assumendo che siano presenti in un sistema almeno due processi utente, quattro possibili ragioni per cui in un moderno sistema operativo time sharing che implementa la memoria virtuale si può verificare un context switch tra due processi utente sono:

Scegli un'alternativa:

a.
1. il processo running esegue una wait e si addormenta sul semaforo
2. il processo running genera una pagina vittima
3. il processo running genera una trap e viene terminato
4. il processo running inizia una operazione di I/O

b.
1. il processo running esegue una wait e si addormenta sul semaforo
2. il processo running genera page fault
3. il processo running genera una trap e viene terminato
4. il processo in coda di ready inizia una operazione di I/O

c.
1. il processo running esegue una wait e si addormenta sul semaforo ✔
2. il processo running genera page fault
3. il processo running genera una trap e viene terminato
4. il processo running inizia una operazione di I/O

d.
1. il processo running esegue una signal e si addormenta sul semaforo
2. il processo running genera page fault
3. il processo running genera una trap e viene terminato
4. il processo running inizia una operazione di I/O

Risposta corretta.

La risposta corretta è:
1. il processo running esegue una wait e si addormenta sul semaforo
2. il processo running genera page fault
3. il processo running genera una trap e viene terminato
4. il processo running inizia una operazione di I/O

Domanda **9**

Risposta errata

Punteggio ottenuto 0,00 su 2,00

Si consideri questo pseudo-codice visto a lezione:

```
Shared data: boolean  lock = false;
Processo Pi:
do {
    while (TestAndSet(&lock));
        sezione critica
        lock = false;
   sezione non critica
   } while (true);
```

Di questo pseudo-codice possiamo dire che:

Scegli un'alternativa:

a. produce uno spreco inutile di tempo di CPU, perché un processo in attesa di entrare in una sezione critica già occupata passa tutto il suo quanto di tempo a testare la variabile di lock.

b. costituisce una soluzione corretta al problema della sezione critica, ha però il difetto di esere basato sul busy-waiting, e dunque di far sprecare inutilmente del tempo di CPU.

c. non garantisce la condizione di progresso, perché ogni processo Pi che tentasse di acquisire il lock quando gli viene assegnata la CPU potrebbe trovare la sezione critica sempre occupata da un altro processo. ✖

d. non garantisce la mutua esclusione, dato che manca l'istruzione per settare a 1 la variabile di lock e dunque segnalare agli altri processi che è stato acquisito il lock e la sezione critica è occupata.

Risposta errata.

La risposta corretta è: produce uno spreco inutile di tempo di CPU, perché un processo in attesa di entrare in una sezione critica già occupata passa tutto il suo quanto di tempo a testare la variabile di lock.

Domanda **10**

Parzialmente corretta

Punteggio ottenuto 2,50 su 3,00

si consideri l'esecuzione della seguente porzione di codice che utilizza la system call fork:

```
int a, b, c, d, n, pid1, pid2, pid3;
a = 20, b = 30, c = 40, d = 50;
n = fork();
      if ( n == 0)
         {a = 25; b = 35;
          pid1 = getppid();
          printf("%d", pid1);
          exit(0);}
      else
         {c = 45; d = 55;
          pid2 = getpid();
          printf("%d",pid2);
          pid3 = wait(NULL);
          exit(0);}
```

il valore della variabile a vista dal processo figlio subito prima della sua exit è [25] ✔

il valore della variabile c vista dal processo figlio subito prima della sua exit è [40] ✔

il valore della variabile b vista dal processo padre subito prima della sua exit è: [30] ✔

il valore della variabile d vista dal processo padre subito prima della sua exit è: [55] ✔

all'esecuzione delle due printf vale la seguente relazione: pid1 [=] ✔ pid2

del risultato della wait possiamo dire che: pid3 [<] ✖ pid1

[35] [=] [25] [>] [55] [40] [45] [<] [30] [50] [20]

Risposta parzialmente esatta.

Hai selezionato correttamente 5.

La risposta corretta è:

si consideri l'esecuzione della seguente porzione di codice che utilizza la system call fork:

```
int a, b, c, d, n, pid1, pid2, pid3;
a = 20, b = 30, c = 40, d = 50;
n = fork();
      if ( n == 0)
         {a = 25; b = 35;
          pid1 = getppid();
          printf("%d", pid1);
          exit(0);}
      else
         {c = 45; d = 55;
          pid2 = getpid();
          printf("%d",pid2);
          pid3 = wait(NULL);
          exit(0);}

il valore della variabile a vista dal processo figlio subito prima della sua exit è [25]
il valore della variabile c vista dal processo figlio subito prima della sua exit è [40]
il valore della variabile b vista dal processo padre subito prima della sua exit è: [30]
il valore della variabile d vista dal processo padre subito prima della sua exit è: [55]
all'esecuzione delle due printf vale la seguente relazione: pid1 [=] pid2
del risultato della wait possiamo dire che: pid3 [>] pid1
```

Domanda **11**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Ricostruite il codice della system call Signal:

signal(semaforo *S) {

S->valore++;

if S->valore <= 0

{

togli un processo P da S->waiting_list;

wakeup(P);

}

}

sleep();

if S->valore < 0

if S->valore == 0

aggiungi questo processo a S->waiting_list;

S->valore--;

Risposta corretta.

La risposta corretta è:
Ricostruite il codice della system call Signal:

signal(semaforo *S) {

[S->valore++;]

[if S->valore <= 0]

{

[togli un processo P da S->waiting_list;]

[wakeup(P);]

}

}