

**Iniziato** martedì, 17 giugno 2025,**Stato** Completato**Terminato** martedì, 17 giugno 2025,**Tempo impiegato****Valutazione** 13,00 su un massimo di 23,00 (56,52%)Domanda **1**

Risposta errata

Punteggio ottenuto 0,00 su 2,00

Di un sistema è noto che la tabella delle pagine più grande del sistema occupa esattamente 2 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 8 Kilobyte.

lo spazio logico del sistema è grande:  ✖lo spazio fisico del sistema è grande:  ✖

Risposta errata.

La risposta corretta è:

Di un sistema è noto che la tabella delle pagine più grande del sistema occupa esattamente 2 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 8 Kilobyte.

lo spazio logico del sistema è grande: [8 Megabyte]

lo spazio fisico del sistema è grande: [256 Gigabyte]

## Domanda 2

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Se si verifica un page fault:

Scegli un'alternativa:

- ☐ a. significa che un processo in coda di ready ha indirizzato una pagina non presente in RAM. Il processo viene tolto dalla coda e il sistema operativo incomincia a recuperare la pagina che era stata indirizzata. A operazione completata il processo viene riportato in coda di ready
- ☐ b. significa che il processo running ha indirizzato una pagina non appartenente al suo spazio di indirizzamento. Il processo viene tolto dall'esecuzione e il sistema operativo incomincia a recuperare la pagina che era stata indirizzata. A operazione completata il processo viene riportato in coda di ready
- ☒ c. significa che il processo running ha indirizzato una pagina non presente in RAM. Il processo viene tolto dall'esecuzione e il sistema operativo incomincia a recuperare la pagina che era stata indirizzata. A operazione completata il processo viene riportato in coda di ready ✓
- ☐ d. significa che il processo running ha indirizzato una pagina non presente in RAM. Il processo viene tolto dall'esecuzione e il sistema operativo incomincia a recuperare la pagina che era stata indirizzata. A operazione completata il processo viene riportato in esecuzione

Risposta corretta.

La risposta corretta è: significa che il processo running ha indirizzato una pagina non presente in RAM. Il processo viene tolto dall'esecuzione e il sistema operativo incomincia a recuperare la pagina che era stata indirizzata. A operazione completata il processo viene riportato in coda di ready

## Domanda 3

Risposta errata

Punteggio ottenuto 0,00 su 2,00

Cosa sono deadlock e starvation? Quale di questi due fenomeni è più grave, e perché?

Scegli un'alternativa:

- ☐ a. Deadlock: due o più processi fermi in una attesa circolare. Starvation: un processo che non riesce a portare avanti la propria computazione. Il deadlock è più grave, perché coinvolge più processi. Inoltre deadlock produce starvation, ma non viceversa
- ☒ b. Deadlock: due o più processi fermi in una attesa circolare. Starvation: un processo che non esce mai dallo stato di wait. Il deadlock è più grave, perché produce starvation. ✗
- ☐ c. Deadlock: due o più processi fermi in attesa di entrare in sezione critica. Starvation: un singolo processo che non riesce mai ad entrare in sezione critica. Il deadlock è più grave, perché coinvolge più processi.
- ☐ d. Starvation: due o più processi fermi in una attesa circolare. Deadlock: un processo che non riesce a portare avanti la propria computazione. La starvation è più grave, perché coinvolge più processi. Inoltre starvation produce deadlock, ma non viceversa

Risposta errata.

La risposta corretta è: Deadlock: due o più processi fermi in una attesa circolare. Starvation: un processo che non riesce a portare avanti la propria computazione. Il deadlock è più grave, perché coinvolge più processi. Inoltre deadlock produce starvation, ma non viceversa

Domanda 4

Risposta corretta

Punteggio ottenuto 2,00 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. 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
- ☒ c.
  - 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
- ☐ d.
  - 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

Risposta corretta.

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 5

Risposta corretta

Punteggio ottenuto 3,00 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 = 60, b = 70, c = 80, d = 90;
n = fork();
    if ( n == 0)
    {a = 65; b = 75;
      pid1 = getpid();
      printf("%d", pid1);
      exit(0);}
    else
    {c = 85; d = 95;
      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  ✓

il valore della variabile c vista dal processo figlio subito prima della sua exit è  ✓

il valore della variabile b vista dal processo padre subito prima della sua exit è:  ✓

il valore della variabile d vista dal processo padre subito prima della sua exit è:  ✓

all'esecuzione delle due printf vale la seguente relazione: pid1  ✓ pid2

del risultato della wait possiamo dire che: pid2  ✓ pid3

Risposta corretta.

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 = 60, b = 70, c = 80, d = 90;
n = fork();
    if ( n == 0)
    {a = 65; b = 75;
      pid1 = getpid();
      printf("%d", pid1);
      exit(0);}
    else
    {c = 85; d = 95;
      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 [65]

il valore della variabile c vista dal processo figlio subito prima della sua exit è [80]

il valore della variabile b vista dal processo padre subito prima della sua exit è: [70]

il valore della variabile d vista dal processo padre subito prima della sua exit è: [95]

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

del risultato della wait possiamo dire che: pid2 [<] pid3


Domanda 6

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Che cosa vuol dire che un algoritmo di scheduling soffre di starvation?

Scegli un'alternativa:

- ☐ a. che non garantisce che, in un tempo finito, un processo possa entrare nella sua sezione critica e portare avanti la sua computazione
- ☐ b. che non garantisce che, in un tempo finito, un processo addormentato su un semaforo possa venire svegliato e portare avanti la sua computazione
- ☒ c. che non garantisce che, in un tempo finito, un processo in coda di ready venga selezionato per entrare in esecuzione e terminare 
- ☐ d. che non garantisce che, in un tempo finito, un processo running non venga più interrotto e possa portare a termine la sua computazione

Risposta corretta.

La risposta corretta è: che non garantisce che, in un tempo finito, un processo in coda di ready venga selezionato per entrare in esecuzione e terminare

## Domanda 7

Risposta errata

Punteggio ottenuto 0,00 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 2
- ☐ b.
  - 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
- ☒ 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: 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

Risposta errata.

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 8

Risposta errata

Punteggio ottenuto 0,00 su 2,00

In hard disk grande 512 Gigabyte, per scrivere il numero di un blocco vengono usati 30 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 32 byte vengono usati per tenere traccia dei blocchi di dati di A. Quanto può essere grande al massimo A?

Scegli un'alternativa:

- ☐ a. 6 Kilobyte
- ☒ b. 8 Kilobyte ❌
- ☐ c. 4 Kilobyte
- ☐ d. 10 Kilobyte

Risposta errata.

La risposta corretta è: 4 Kilobyte

## Domanda 9

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Su un hard disk che adotta una allocazione concatenata (senza FAT) è memorizzato un file A della dimensione di 0x8000 byte, e si sa che nell'ultimo blocco di A sono presenti 32 byte del file. Si sa inoltre che per scrivere il numero di un blocco vengono usati 28 bit, arrotondati al minimo numero di byte necessario. Quanto è grosso l'hard disk?

Scegli un'alternativa:

- ☐ a. 2 Terabyte
- ☐ b. 512 Gigabyte
- ☐ c. 4 Terabyte
- ☒ d. 1 Terabyte ✔️

Risposta corretta.

La risposta corretta è: 1 Terabyte

Domanda **10**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

In quali casi e perché si usa una Inverted Page Table?

Scegli un'alternativa:

- ☒ a. Le IPT si usano nei sistemi con spazi di indirizzamento logici molto grandi per evitare la paginazione a più livelli, che rende la traduzione degli indirizzi da logici a fisici molto onerosa quando non è possibile ricorrere al TLB ✓
- ☐ b. Le IPT si usano quando abbiamo degli spazi di indirizzamento logico molto piccoli, perché in questo modo possiamo usare un'unica page table per tutti i processi, anziché averne una per ciascun processo
- ☐ c. Le IPT si usano nei sistemi con spazi di indirizzamento fisico piccoli, in questo modo viene sprecato meno spazio da parte delle page table dei processi, attraverso la paginazione a più livelli
- ☐ d. Le IPT si usano quando non è disponibile il TLB, e dunque la traduzione degli indirizzi da logici a fisici risulterebbe troppo onerosa usando delle normali page table

Risposta corretta.

La risposta corretta è: Le IPT si usano nei sistemi con spazi di indirizzamento logici molto grandi per evitare la paginazione a più livelli, che rende la traduzione degli indirizzi da logici a fisici molto onerosa quando non è possibile ricorrere al TLB



Domanda **11**

Risposta errata

Punteggio ottenuto 0,00 su 2,00

Ricostruite il codice del generico produttore nel problema dei produttori-consumatori:

full = 0 ; empty = SIZE; mutex = 1;

produttore:

while (true) {

    produci un item in nextp

        wait(mutex);

✗

        if empty == SIZE wait(full);

✗

    inserisci un elemento nel buffer

        wait(empty);

✗

        if full == 0 signal(empty);

✗

}

signal(empty);

wait(full);

signal(mutex);

if full == SIZE wait(full);

signal(full);

Risposta errata.

La risposta corretta è:

Ricostruite il codice del generico produttore nel problema dei produttori-consumatori:

full = 0 ; empty = SIZE; mutex = 1;

produttore:

while (true) {

    produci un item in nextp

    [wait(empty);]

    [wait(mutex);]

    inserisci un elemento nel buffer

    [signal(mutex);]

    [signal(full);]

}