

**Iniziato** martedì, 27 gennaio 2026, 09:06**Stato** Completato**Terminato** martedì, 27 gennaio 2026, 10:05**Tempo impiegato** 59 min. 52 secondi**Valutazione** 28,00 su un massimo di 30,00 (93,33%)Domanda **1**

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 page fault
  - 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 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
- ☐ 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 in coda di ready inizia una operazione di I/O
- ☐ d.
  - 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

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 2

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Un sistema ha un tempo di accesso in RAM di 120 ns, adotta un TLB con un tempo di accesso di 10 ns e un hit rate del 95%, usa una paginazione a due livelli e non ha bisogno di usare un algoritmo di rimpiazzamento delle pagine. Qual è il tempo medio di accesso alla RAM (*medium access time - mat*) del sistema? (per semplicità in caso di miss si ignori il costo di interrogazione del TLB)

Scegli un'alternativa:

- ☐ a. mat = 135,5 ns
- ☒ b. mat = 141,5 ns ✓
- ☐ c. mat = 147,5 ns
- ☐ d. mat = 130,5 ns

Risposta corretta.

La risposta corretta è: mat = 141,5 ns

## Domanda 3

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

In un sistema che adotta la paginazione della memoria, un indirizzo logico è scritto su "M" bit, e il numero di pagina è scritto su "l" bit. Lo spazio fisico è invece suddiviso in  $2^{32}$  frame.

Possiamo quindi dire che:

un frame del sistema è grande:  $2^{(M-l)}$  ✓ bytelo spazio fisico del sistema è grande:  $2^{(32+M-l)}$  ✓ bytela tabella delle pagine più grande del sistema ha una dimensione di:  $(2^l) * 4$  ✓ byteil sistema dovrà implementare la memoria virtuale solo se:  $M > (32+M-l)$  ✓

$2^{(32-l)}$

$M < (32+M-l)$

$2^{(M+32)}$

$(2^{32}) * 2$

$2^{(32-M+l)}$

Risposta corretta.

La risposta corretta è:

In un sistema che adotta la paginazione della memoria, un indirizzo logico è scritto su "M" bit, e il numero di pagina è scritto su "l" bit. Lo spazio fisico è invece suddiviso in  $2^{32}$  frame.

Possiamo quindi dire che:

un frame del sistema è grande:  $[2^{(M-l)}]$  bytelo spazio fisico del sistema è grande:  $[2^{(32+M-l)}]$  bytela tabella delle pagine più grande del sistema ha una dimensione di:  $[(2^l) * 4]$  byteil sistema dovrà implementare la memoria virtuale solo se:  $[M > (32+M-l)]$

## Domanda 4

Risposta errata

Punteggio ottenuto 0,00 su 2,00

Con la paginazione della memoria si ottiene una forma automatica di protezione dello spazio di indirizzamento di ciascun processo. Dite se l'affermazione è vera o falsa, e spiegate perché.

Scegli un'alternativa:

- ☒ a. l'affermazione è vera: qualsiasi indirizzo logico usato da un processo P è scritto su un numero di bit che determina la dimensione dello spazio di indirizzamento logico usato da P, e quindi quel processo può indirizzare solo frame appartenenti al processo stesso ✗
- ☐ b. l'affermazione è falsa: la protezione dello spazio di indirizzamento dei processi è compito del sistema operativo, ed è svolta attraverso opportuni meccanismi hardware, in particolare attraverso il registro di rilocalizzazione
- ☐ c. l'affermazione è falsa: la protezione dello spazio di indirizzamento dei processi è compito del sistema operativo, ed è svolta attraverso opportuni meccanismi software, che verificano che ogni indirizzo usato da un processo non acceda aree di RAM non in uso al processo stesso
- ☐ d. l'affermazione è vera: qualsiasi indirizzo logico usato da un processo P può indirizzare solo i frame che contengono pagine appartenenti allo spazio logico di P, e l'offset specificato nell'indirizzo logico non può che specificare uno qualsiasi dei byte del frame indirizzato

Risposta errata.

La risposta corretta è: l'affermazione è vera: qualsiasi indirizzo logico usato da un processo P può indirizzare solo i frame che contengono pagine appartenenti allo spazio logico di P, e l'offset specificato nell'indirizzo logico non può che specificare uno qualsiasi dei byte del frame indirizzato

## Domanda 5

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Tra gli **svantaggi** dell'allocazione concatenata nella variante della FAT troviamo:

Scegli un'alternativa:

- ☒ a. che fa lavorare di più il sistema operativo perché deve essere periodicamente salvata sull'hard disk e sottrae spazio ai processi utente in memoria primaria ✓
- ☐ b. che fa sprecare spazio in memoria primaria per tenere traccia di file molto piccoli e deve essere periodicamente salvata nell'area di swap
- ☐ c. che occupa spazio in memoria primaria ed è inefficiente per tenere traccia dei blocchi liberi dell'hard disk
- ☐ d. che occupa spazio in memoria secondaria e se viene persa, l'accesso sequenziale ai file rimane possibile ma quello diretto diviene molto inefficiente.

Risposta corretta.

La risposta corretta è: che fa lavorare di più il sistema operativo perché deve essere periodicamente salvata sull'hard disk e sottrae spazio ai processi utente in memoria primaria


Domanda 6

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

L'allocazione indicizzata dello spazio in memoria secondaria è particolarmente **svantaggiosa** quando:

Scegli un'alternativa:

- ☐ a. nel caso di file molto grandi, perché la perdita del blocco indice produce la perdita dell'intero file
- ☐ b. nel caso di file che devono essere acceduti in modo sequenziale, in quanto l'allocazione indicizzata è progettata esplicitamente per favorire l'accesso diretto ai dati del file
- ☒ c. quando si deve allocare spazio per file grandi meno di un blocco di dati, perché un intero blocco indice viene usato per tenere traccia di quell'unico blocco, e quindi quasi tutto il blocco indice rimane inutilizzato 
- ☐ d. in nessun caso, infatti l'allocazione indicizzata, nelle sue varianti commerciali, è proprio quella impiegata sia nei sistemi Unix che nei sistemi Windows

Risposta corretta.

La risposta corretta è: quando si deve allocare spazio per file grandi meno di un blocco di dati, perché un intero blocco indice viene usato per tenere traccia di quell'unico blocco, e quindi quasi tutto il blocco indice rimane inutilizzato

## Domanda 7

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Ricostruite il codice della system call Signal:

signal(semaforo \*S) {

S-&gt;valore + + ;



if S-&gt;valore &lt; = 0



{

togli un processo P da S-&gt;waiting\_list;



wakeup(P);



}

}

aggiungi questo processo a S-&gt;waiting\_list;

if S-&gt;valore &lt; 0

sleep();

S-&gt;valore--;

if S-&gt;valore == 0

Risposta corretta.

La risposta corretta è:

Ricostruite il codice della system call Signal:

signal(semaforo \*S) {

[S-&gt;valore + + ;]

[if S-&gt;valore &lt; = 0]

{

[togli un processo P da S-&gt;waiting\_list;]

[wakeup(P);]

}

}

Domanda 8

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Un hard disk ha la dimensione di 512 Gigabyte, è formattato in blocchi da 0x400 byte e adotta una qualche forma di allocazione indicizzata dello spazio su disco. Sull'hard disk è memorizzato un file A della dimensione di 600 Kbyte. Quante operazioni di I/O sono necessarie per leggere l'ultimo blocco di dati del file, assumendo già in RAM tutti gli attributi del file? (selezionate l'opzione di risposta che riporta il ragionamento numerico corretto)

Scegli un'alternativa:

- ☒ a.  $2^{39}/2^{10} = 2^{29}$ , dunque un blocco indice di questo hard disk può tenere traccia di un massimo di 256 Kbyte di dati. Assumendo una allocazione indicizzata a schema concatenato, ci vogliono 3 blocchi indice per registrare l'ultimo blocco di dati del file, e dunque, se è già in RAM il numero del primo blocco indice, saranno necessarie in tutto 4 operazioni di lettura su disco ✓
- ☐ b.  $2^{39}/2^{10} = 2^{29}$ , dunque un blocco indice di questo hard disk può tenere traccia di un massimo di 256 Kbyte di dati. Assumendo una allocazione indicizzata a più livelli, ci vogliono 3 livelli di indizione per registrare l'ultimo blocco di dati del file, e dunque, se è già in RAM il numero del blocco indice più esterno, saranno necessarie in tutto 4 operazioni di lettura su disco
- ☐ c.  $2^{39}/2^{11} = 2^{28}$ , dunque un blocco indice di questo hard disk può tenere traccia di un massimo di 512 Kbyte di dati. Assumendo una allocazione indicizzata a più livelli, ci vogliono 3 livelli di indizione per registrare l'ultimo blocco di dati del file, e dunque, se è già in RAM il numero blocco indice più esterno, saranno necessarie in tutto 4 operazioni di lettura su disco
- ☐ d.  $2^{39}/2^{11} = 2^{28}$ , dunque un blocco indice di questo hard disk può tenere traccia di un massimo di 512 Kbyte di dati. Assumendo una allocazione indicizzata a schema concatenato, ci vogliono 2 blocchi indice per registrare l'ultimo blocco di dati del file, e dunque, se è già in RAM il numero del primo blocco indice, saranno necessarie in tutto 3 operazioni di lettura su disco

Risposta corretta.

La risposta corretta è:  $2^{39}/2^{10} = 2^{29}$ , dunque un blocco indice di questo hard disk può tenere traccia di un massimo di 256 Kbyte di dati. Assumendo una allocazione indicizzata a schema concatenato, ci vogliono 3 blocchi indice per registrare l'ultimo blocco di dati del file, e dunque, se è già in RAM il numero del primo blocco indice, saranno necessarie in tutto 4 operazioni di lettura su disco

Domanda 9

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

In un sistema operativo che adotta uno scheduling con diritto di prelazione, quattro processi arrivano al tempo indicato e consumano la quantità di CPU indicata nella tabella sottostante

Processo	T. di arrivo	Burst
Pa	0	7
Pb	2	4
Pc	4	1
Pd	6	8

Qual è il waiting time medio ottenuto per lo scheduling dei quattro processi della tabella se si usa l'algoritmo di scheduling preemptive che fornisce il miglior turnaround time possibile? Qual è il corrispondente diagramma di GANTT?

Scegli un'alternativa:

- ☐ a. Diagramma di GANTT: (0)...Pa...(2)...Pb...(6)...Pc...(7)...Pa...(10)...Pb...(12)...Pd...(20)  
Waiting time medio = 4
- ☒ b. Diagramma di GANTT: (0)...Pa...(2)...Pb...(4)...Pc...(5)...Pb...(7)...Pa...(12)...Pd...(20) ✓  
Waiting time medio = 3
- ☐ c. Diagramma di GANTT: (0)...Pa...(2)...Pb...(4)...Pc...(5)...Pa...(10)...Pb...(12)...Pd...(20)  
Waiting time medio = 3
- ☐ d. Diagramma di GANTT: (0)...Pa...(2)...Pc...(3)...Pb...(7)...Pa...(10)...Pb...(12)...Pd...(20)  
Waiting time medio = 4

Risposta corretta.

La risposta corretta è:

Diagramma di GANTT: (0)...Pa...(2)...Pb...(4)...Pc...(5)...Pb...(7)...Pa...(12)...Pd...(20)

Waiting time medio = 3

Domanda **10**

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 ../newfolder folder2
7: ln -s 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: 2
  - 2. il link counter di *tmp* è: aumentato di 1
  - 3. l'output del comando 10 è: "ciao" seguito da "salve"
  - 4. il comando 6 da come risultato: un errore perché non sono ammessi hard link tra cartelle
- ☒ b.
  - 1. il link-counter dell'i-node di *paperino* è: 1 ✓
  - 2. il link counter di *tmp* è: aumentato di 2
  - 3. l'output del comando 10 è: "ciao" seguito da "salve"
  - 4. il comando 6 da come risultato: un errore perché non sono ammessi hard link tra cartelle
- ☐ c.
  - 1. il link-counter dell'i-node di *paperino* è: 1
  - 2. il link counter di *tmp* è: aumentato di 2
  - 3. l'output del comando 10 è: "ciao"
  - 4. il comando 6 da come risultato: un nuovo collegamento alla cartella newfolder
- ☐ d.
  - 1. il link-counter dell'i-node di *paperino* è: 2
  - 2. il link counter di *tmp* è: 2
  - 3. l'output del comando 10 è: no such file or directory
  - 4. il comando 6 da come risultato: un errore perché non sono ammessi hard link tra cartelle

Risposta corretta.

La risposta corretta è:

- 1. il link-counter dell'i-node di *paperino* è: 1
- 2. il link counter di *tmp* è: aumentato di 2
- 3. l'output del comando 10 è: "ciao" seguito da "salve"
- 4. il comando 6 da come risultato: un errore perché non sono ammessi hard link tra cartelle



Domanda **11**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Un algoritmo di scheduling a code multiple con retroazione:

Scegli un'alternativa:

- ☐ a. usa più code di ready, gestite ciascuna con una diversa politica di scheduling. Un processo può essere promosso a una coda con priorità superiore se ha consumato completamente il suo ultimo quanto di tempo
- ☐ b. usa più code di ready, gestite ciascuna con una diversa politica di scheduling. Un processo può essere retrocesso a una coda con priorità inferiore se non ha consumato completamente il suo ultimo quanto di tempo
- ☒ c. usa più code di ready, gestite ciascuna con una diversa politica di scheduling. Un processo può essere spostato da una coda all'altra in base a come si è comportato l'ultima volta che gli è stata assegnata la CPU ✓
- ☐ d. usa più code di ready, in cui i processi vengono inseriti a seconda che siano processi CPU o I/O bound, e processi in foreground o in background

Risposta corretta.

La risposta corretta è: usa più code di ready, gestite ciascuna con una diversa politica di scheduling. Un processo può essere spostato da una coda all'altra in base a come si è comportato l'ultima volta che gli è stata assegnata la CPU

## Domanda 12

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

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 = 30, b = 40, c = 50, d = 60;
n = fork();
    if ( n == 0)
        {a = 35; b = 45;
          pid1 = getpid();
          printf("%d", pid1);
          exit(0);}
    else
        {c = 55; d = 65;
          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: pid1  ✓ 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 = 30, b = 40, c = 50, d = 60;
n = fork();
    if ( n == 0)
        {a = 35; b = 45;
          pid1 = getpid();
          printf("%d", pid1);
          exit(0);}
    else
        {c = 55; d = 65;
          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 [35]

il valore della variabile c vista dal processo figlio subito prima della sua exit è [50]

il valore della variabile b vista dal processo padre subito prima della sua exit è: [40]

il valore della variabile d vista dal processo padre subito prima della sua exit è: [65]

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

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

## Domanda 13

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Una Inverted Page Table può essere così descritta:

Scegli un'alternativa:

- ☐ a. ha un numero di entry pari al numero di frame in cui è suddiviso lo spazio fisico del processo in esecuzione, e ogni entry contiene il numero di una pagina. Per tradurre un indirizzo logico P-offset in un indirizzo fisico si cerca P nelle entry della IPT. Se si trova la coppia alla entry F-esima, allora l'indirizzo fisico è costituito dalla coppia "F-offset"
- ☐ b. ha un numero di entry pari al numero di pagine in cui è suddiviso lo spazio logico di un processo, e ogni entry contiene il numero di un frame. Ogni indirizzo logico è formato dalla tripla "PID-numero di pagina-offset". Per tradurre un indirizzo da logico a fisico si cerca il numero della pagina nelle entry della IPT. Se si trova la coppia alla entry F-esima, allora l'indirizzo fisico è costituito dalla coppia "F-offset"
- ☐ c. ha un numero di entry pari al numero di pagine in cui è suddiviso lo spazio logico di un processo, e ogni entry contiene una coppia "PID-frame". Ogni indirizzo logico è formato dalla tripla "PID-numero di frame-offset". Per tradurre un indirizzo da logico a fisico si cerca la coppia "PID-numero di frame" nelle entry della IPT. Se si trova la coppia alla entry F-esima, allora l'indirizzo fisico è costituito dalla coppia "F-offset"
- ☒ d. ha un numero di entry pari al numero di frame in cui è suddivisa la memoria fisica, e ogni entry contiene una coppia "PID-numero di pagina". Ogni indirizzo logico è formato dalla tripla "PID-numero di pagina-offset". Per tradurre un indirizzo da logico a fisico si cerca la coppia "PID-numero di pagina" nelle entry della IPT. Se si trova la coppia alla entry F-esima, allora l'indirizzo fisico è costituito dalla coppia "F-offset" ✓

Risposta corretta.

La risposta corretta è: ha un numero di entry pari al numero di frame in cui è suddivisa la memoria fisica, e ogni entry contiene una coppia "PID-numero di pagina". Ogni indirizzo logico è formato dalla tripla "PID-numero di pagina-offset". Per tradurre un indirizzo da logico a fisico si cerca la coppia "PID-numero di pagina" nelle entry della IPT. Se si trova la coppia alla entry F-esima, allora l'indirizzo fisico è costituito dalla coppia "F-offset"

## Domanda 14

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Un hard disk ha la dimensione di 512 Gigabyte ed è formattato in blocchi da 0x1000 byte. Qual è la dimensione della FAT dell'hard disk? (selezionate l'opzione di risposta che riporta l'espressione aritmetica corretta "num. entry \* dim. entry" per il calcolo della dimensione della FAT)

Scegli un'alternativa:

- ☐ a.  $2^{26} * 2^3 = 512$  Megabyte
- ☒ b.  $2^{27} * 2^2 = 512$  Megabyte ✓
- ☐ c.  $2^{26} * 2^2 = 256$  Megabyte
- ☐ d.  $2^{27} * 2 = 256$  Megabyte

Risposta corretta.

La risposta corretta è:  $2^{27} * 2^2 = 512$  Megabyte

Domanda **15**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

In un sistema paginato è noto che lo spreco di memoria primaria dovuto alla frammentazione interna è in media di circa 1 Kbyte per processo. Un indirizzo fisico è scritto su 26 bit e lo spazio di indirizzamento logico è 4 volte quello fisico.

Qual è la dimensione della tabella delle pagine più grande di questo sistema? (selezionate l'opzione di risposta che riporta il ragionamento aritmetico e il risultato corretti)

Scegli un'alternativa:

- ☐ a. Ogni entry della PT più grande del sistema deve essere grande almeno 3 byte, e dunque la PT sarà grande  $3 * 2^{15} = 96$  Kbyte (circa)
- ☐ b. Ogni entry della PT più grande del sistema deve essere grande almeno 2 byte, e dunque la PT sarà grande  $2 * 2^{15} = 64$  Kbyte (circa)
- ☒ c. Ogni entry della PT più grande del sistema deve essere grande almeno 2 byte, e dunque la PT sarà grande  $2 * 2^{17} = 256$  Kbyte (circa) ✓
- ☐ d. Ogni entry della PT più grande del sistema deve essere grande almeno 3 byte, e dunque la PT sarà grande  $3 * 2^{17} = 384$  Kbyte (circa)

Risposta corretta.

La risposta corretta è: Ogni entry della PT più grande del sistema deve essere grande almeno 2 byte, e dunque la PT sarà grande  $2 * 2^{17} = 256$  Kbyte (circa)