Iniziato	giovedì, 12 febbraio 2026, 11:02
Stato	Completato
Terminato	giovedì, 12 febbraio 2026, 11:47
Tempo impiegato	44 min. 31 secondi
Valutazione	**22,00** su un massimo di 30,00 (**73,33**%)

Domanda **1**

Risposta errata

Punteggio ottenuto 0,00 su 2,00

Secondo quanto visto a lezione, quale/quali dei seguenti comandi Unix modifica il valore del *link counter* dell'index-node associato al file di testo X? (si assuma di avere i permessi per eseguire tutti i comandi e di essere posizionati in una generica cartella user/tmp che contiene X)

1) ln -s X Y

2) ls X Y

3) ln X ../X

4) rm X Y

Scegli un'alternativa:

a. i comandi 3) e 4)

b. i comandi 1) e 4)

c. il comando 4) ✖

d. i comandi 1), 3) e 4)

Risposta errata.

La risposta corretta è: i comandi 3) e 4)

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%, e usa un algoritmo di rimpiazzamento delle pagine. Quando si verifica un hit la pagina indirizzata è sicuramente in RAM. Quando si verifica un miss, nel 20 % dei casi la pagina indirizzata non è in RAM e il page fault ha un costo totale di gestione di 1 microsecondo, indipendentemente dal valore del dirty bit. Qual è l'effective access time (*eat*) del sistema? (per semplicità in caso di miss si ignori il costo di interrogazione del TLB, e in caso di page fault si consideri solo il tempo di gestione del page fault)

Scegli un'alternativa:

a. mat = 143,1 ns ✔

b. mat = 150,1 ns

c. mat = 133,1 ns

d. mat = 137,1 ns

Risposta corretta.

La risposta corretta è: mat = 143,1 ns

Domanda **3**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Confrontando le librerie statiche e quelle dinamiche possiamo dire che:

Scegli un'alternativa:

a. le librerie dinamiche sono sicuramente preferibili a quelle statiche, ad esempio perché vengono caricate in RAM solo se effettivamente usate dai processi, e al contrario di quelle statiche possono essere usate col codice dinamicamente rilocabile

b. le librerie dinamiche sono sicuramente preferibili a quelle statiche, ad esempio perché al contrario di quelle statiche possono essere usate con la memoria virtuale, e in caso di aggiornamento non costringono alla ricompilazione dei processi che le usano

c. le librerie dinamiche sono sicuramente preferibili a quelle statiche, ad esempio perché vengono caricate in RAM solo se effettivamente usate dai processi, e perché quelle statiche possono essere usate solo con il codice statico.

d. le librerie dinamiche sono sicuramente preferibili a quelle statiche, ad esempio perché vengono caricate in RAM solo se effettivamente usate dai processi, e in caso di aggiornamento non costringono alla ricompilazione dei processi che le usano ✔

Risposta corretta.

La risposta corretta è: le librerie dinamiche sono sicuramente preferibili a quelle statiche, ad esempio perché vengono caricate in RAM solo se effettivamente usate dai processi, e in caso di aggiornamento non costringono alla ricompilazione dei processi che le usano

Domanda **4**

Risposta errata

Punteggio ottenuto 0,00 su 2,00

In un sistema operativo moderno che implementa la memoria virtuale, quando si verifica un page fault e non c'è spazio in RAM occorre scegliere una pagina vittima. Dei criteri di scelta visti a lezione possiamo dire che:

Scegli un'alternativa:

a. scegliere la pagina che verrà riferita più in là nel tempo è sicuramente il criterio migliore, ma richiede hardware dedicato offerto solo dalle CPU di fascia alta

b. scegliere la pagina che è entrata in RAM da più tempo è un buon criterio, e infatti viene adottato da molti sistemi operativi ✖

c. scegliere una pagina che è stata modificata di recente è un buon criterio, nel caso in cui ci sia bisogno di salvare periodicamente lo stato della computazione del processo

d. scegliere la pagina che è stata riferita da più tempo è un ottimo criterio, ma all'atto pratico non si può implementare in modo efficiente

Risposta errata.

La risposta corretta è: scegliere la pagina che è stata riferita da più tempo è un ottimo criterio, ma all'atto pratico non si può implementare in modo efficiente

Domanda **5**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

In un sistema operativo un indirizzo fisico è scritto su 28 bit, l'offset più grande in una pagina è 3FFF, e lo spazio logico è il doppio di quello fisico.

Se il sistema adottasse una Inverted Page Table della dimensione di 64 Kilobyte, quanti potrebbero essere al massimo i processi presenti contemporaneamente nel sistema?

Scegli un'alternativa:

a. 2^20 processi

b. 2^14 processi

c. 2^17 processi ✔

d. 2^10 processi

Risposta corretta.

La risposta corretta è: 2^17 processi

Domanda **6**

Risposta corretta

Punteggio ottenuto 2,00 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. 4 Kilobyte ✔
- b. 10 Kilobyte
- c. 8 Kilobyte
- d. 6 Kilobyte

Risposta corretta.

La risposta corretta è: 4 Kilobyte

Domanda **7**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

GESTIONE DEI PROCESSI

Si consideri questa variante del problema dei produttori e consumatori:

Semaphore mutex = 1; full =0; empty =N;

codice consumatore:

```
repeat
wait(full);
<preleva dato dal buffer>
signal(empty)
<consuma dato>
forever
```

codice produttore:

```
repeat
 <produci dato>
 wait(empty)
 wait(mutex);
 <inserisci dato nel buffer>
 signal(mutex);
 signal(full)
 forever
```

Scegli un'alternativa:

a. La soluzione proposta non funziona perchè se ci sono almeno due posizioni libere nel buffer, ossia se emtpy >= 2, ci potrebbero essere almeno 2 consumatori che superano la wait su empty e tentano di operare non in mutua esclusione sul buffer condiviso.

b. La soluzione proposta funziona a condizione che sia presente un solo consumatore, e non è quindi necessario proteggere con una sezione critica il prelievo di un elemento dal buffer condiviso ✔

c. La soluzione proposta non funziona perchè se ci sono almeno due dati nel buffer, ossia se full >= 2, ci potrebbero essere almeno 2 produttori che superano la wait su full e tentano di operare non in mutua esclusione sul buffer condiviso.

d. La soluzione proposta funziona a condizione che i consumatori accedano al buffer condiviso i momenti diversi. Manca infatti la mutua esclusione rispetto all'operazione di prelievo di un elemento dal buffer condiviso

Risposta corretta.

La risposta corretta è: La soluzione proposta funziona a condizione che sia presente un solo consumatore, e non è quindi necessario proteggere con una sezione critica il prelievo di un elemento dal buffer condiviso

Domanda **8**

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: echo "ciao" > pippo // crea un nuovo file di nome *pippo* contenente la stringa *ciao*
4: cd newfolder
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: 3
   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: 2
   2. il link counter di *newfolder* è: 3
   3. l'output del comando 10 è: "ciao"
   4. il link counter di tmp è: aumentato di 1

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: 3
   3. l'output del comando 10 è: no such file or directory
   4. il link counter di tmp è: aumentato di 1

Risposta errata.

La risposta corretta è:

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

Domanda **9**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

I semafori sono sezioni critiche che possono essere implementate sfruttando la disabilitazione degli interrupt. E' ragionevole usare questa soluzione anche per implementare le sezioni critiche dei processi utente?

Scegli un'alternativa:

a. non è una soluzione ragionevole perché in questo modo un solo processo alla volta potrebbe usare la wait e la signal per accedere ad una certa sezione critica

b. è una soluzione ragionevole, perché in questo modo il processo in sezione critica può portare a termine la sua computazione senza venire interrotto a metà

c. è una soluzione ragionevole perché permette di evitare l'uso di wait e signal, che sono primitive comunque costose da implementare

d. non è una soluzione ragionevole perché disabilitare gli interrupt toglierebbe al sistema operativo il controllo della macchina ✔

Risposta corretta.

La risposta corretta è: non è una soluzione ragionevole perché disabilitare gli interrupt toglierebbe al sistema operativo il controllo della macchina

Domanda **10**

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 2 byte usando usando tutti i bit a disposizione, e nel sistema sono presenti in media 4 processi che insieme producono una frammentazione interna complessiva media di 4 Kilobyte.

lo spazio logico del sistema è grande: 8 Megabyte ✖

lo spazio fisico del sistema è grande: 256 Megabyte ✖

	nessuno dei valori proposti
64 Megabyte	non si può ricavare dai dati del problema
128 Megabyte	2 Megabyte
4 Megabyte	

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 2 byte usando usando tutti i bit a disposizione, e nel sistema sono presenti in media 4 processi che insieme producono una frammentazione interna complessiva media di 4 Kilobyte.

lo spazio logico del sistema è grande: [4 Megabyte]
lo spazio fisico del sistema è grande: [128 Megabyte]

Domanda **11**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

In un sistema operativo che adotta uno scheduling senza 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	8
Pb	2	8
Pc	4	2
Pd	6	1

Se si usa l'algoritmo di scheduling non preemptive che fornisce le migliori prestazioni possibili per schedulare i 4 processi in tabella:

il waiting time medio è: 4 ✔

il turnaround medio è: 35/4 ✔

il diagramma di GANTT è: (0) ... Pa ... (8) ... Pd ... (9) ... Pc ... (11) ... Pb ... (19) ✔

l'algoritmo usato per le risposte precedenti può soffrire di starvation?

si ✔

5

(0) ... Pa ... (8) ... Pc ... (10) ... Pb ... (18) ... Pd ... (19)

36/4

no

Risposta corretta.

La risposta corretta è:

In un sistema operativo che adotta uno scheduling senza 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	8
Pb	2	8
Pc	4	2
Pd	6	1

Se si usa l'algoritmo di scheduling non preemptive che fornisce le migliori prestazioni possibili per schedulare i 4 processi in tabella:

il waiting time medio è: [4]

il turnaround medio è: [35/4]

il diagramma di GANTT è: [(0) ... Pa ... (8) ... Pd ... (9) ... Pc ... (11) ... Pb ... (19)]

l'algoritmo usato per le risposte precedenti può soffrire di starvation? [si]

Domanda **12**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Si consideri il diagramma di transizione degli stati di un processo che descrive un sistema multi-tasking. Come è possibile modificare il diagramma per descrivere un sistema time-sharing?

Scegli un'alternativa:

a. basta aggiungere un arco che vada dallo stato “waiting” allo stato “ready to run", che rappresenta il caso in cui il processo ha completato una operazione di I/O e può tornare a competere per l'uso della CPU

b. basta aggiungere un arco che vada dallo stato “running” allo stato “waiting", che rappresenta il generico caso in cui il processo in esecuzione abbandona la CPU perché deve compiere una operazione di I/O

c. basta aggiungere un arco che vada dallo stato “running” allo stato “ready to run", che rappresenta il caso in cui il processo in esecuzione viene rimesso in coda di ready perché è scaduto il suo quanto di tempo ✔

d. basta aggiungere un arco che vada dallo stato “ready to run” allo stato “running", che rappresenta il caso in cui il processo è arrivato in testa alla coda di ready e quindi è il suo turno di entrare in esecuzione

Risposta corretta.

La risposta corretta è: basta aggiungere un arco che vada dallo stato “running” allo stato “ready to run", che rappresenta il caso in cui il processo in esecuzione viene rimesso in coda di ready perché è scaduto il suo quanto di tempo

Domanda **13**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Dell'allocazione indicizzata possiamo dire che:

Scegli un'alternativa:

a. è particolarmente adatta per hard disk di dimensioni limitate, per allocare file piccoli spreca molti byte, fornisce un accesso diretto ai file efficiente.

b. è particolarmente adatta per hard disk di dimensioni limitate, per allocare file piccoli spreca molti byte, fornisce un accesso sequenziale inefficiente ai file molto grandi.

c. è la più adottata nei sistemi operativi moderni, per allocare file piccoli spreca molti byte, fornisce un accesso sequenziale inefficiente ai file molto grandi.

d. è la più adottata nei sistemi operativi moderni, per allocare file piccoli spreca molti byte, fornisce un accesso diretto ai file efficiente. ✔

Risposta corretta.

La risposta corretta è: è la più adottata nei sistemi operativi moderni, per allocare file piccoli spreca molti byte, fornisce un accesso diretto ai file efficiente.

Domanda **14**

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 0x4000 byte, e si sa che nell'ultimo blocco di A sono presenti 8 byte del file. Si sa inoltre che per scrivere il numero di un blocco vengono usati 29 bit, arrotondati al minimo numero di byte necessario. Quanto è grosso l'hard disk?

Scegli un'alternativa:

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

Risposta corretta.

La risposta corretta è: 4 Terabyte

Domanda **15**

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 = 10, b = 20, c = 30, d = 40;
n = fork();
      if ( n == 0)
         {a = 15; b = 25;
          pid1 = getppid();
          printf("%d", pid1);
          exit(0);}
      else
         {c = 35; d = 45;
          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 [15] ✔

il valore della variabile c vista dal processo figlio subito prima della sua exit è [30] ✔

il valore della variabile b vista dal processo padre subito prima della sua exit è: [20] ✔

il valore della variabile d vista dal processo padre subito prima della sua exit è: [45] ✔

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

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

[35] [20] [30] [10] [=] [45] [>] [40] [<] [25] [15]

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 = 10, b = 20, c = 30, d = 40;
n = fork();
      if ( n == 0)
         {a = 15; b = 25;
          pid1 = getppid();
          printf("%d", pid1);
          exit(0);}
      else
         {c = 35; d = 45;
          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 [15]
il valore della variabile c vista dal processo figlio subito prima della sua exit è [30]
il valore della variabile b vista dal processo padre subito prima della sua exit è: [20]
il valore della variabile d vista dal processo padre subito prima della sua exit è: [45]
all'esecuzione delle due printf vale la seguente relazione: pid1 [=] pid2
del risultato della wait possiamo dire che: pid3 [>] pid1
```