

# Sistemi Operativi 2015.09.24 (computer pari)

[Home](#) ► [Courses](#) ► [Miscellaneous](#) ► [SOpari20150924](#) ► [General](#) ► [Compito pari - turno 1 - 6 cfu](#) ► [Preview](#)

## Quiz navigation

[1](#) [2](#) [3](#) [4](#) [i](#) [5](#)

[6](#) [7](#) [8](#) [9](#)

[Finish attempt ...](#)

[Start a new preview](#)

## Administration

### Quiz administration

- [Edit settings](#)
- [Group overrides](#)
- [User overrides](#)

### Edit quiz

### Preview

- [Results](#)
- [Locally assigned roles](#)
- [Permissions](#)
- [Check permissions](#)
- [Filters](#)
- [Logs](#)
- [Backup](#)
- [Restore](#)
- [Question bank](#)

### Course administration

[Switch role to...](#)

[My profile settings](#)

### Site administration

[Search](#)

## Question 1

Not yet answered

Not graded

[Flag question](#)

[Edit question](#)

## Dati studente

Inserisci qui i tuoi dati, **compila subito questa parte.**

Quanti CFU?

5 cfu

6 cfu

Cognome

Nome

Matricola

email

Numero Computer

Ordinamento (509, 270, erasmus, ecc.)

## Question 2

Not yet answered

Not graded

[Flag question](#)

[Edit question](#)

## Memory management

Rispondi, **brevemente e punto per punto**, a ciascuna delle seguenti domande circa il fenomeno del trashing.

1. In cosa consiste il fenomeno chiamato trashing?
2. Una buona strategia di eviction può evitare che il sistema vada in trashing? Perché?
3. Un sistema con disk cache è più o meno facile che vada in trashing rispetto ad un sistema senza disk cache? Perché?
4. Un sistema con shared libraries è più o meno facile che vada in trashing rispetto ad un sistema senza shared libraries? Perché?

Paragraph

1.

2.

3.

4.

Path: p

## Question 3

Not yet answered

Not graded

[Flag question](#)

## I/O

In una operazione di **lettura da disco**, quali sono gli **eventi e/o comportamenti** che coinvolgono il processo, il disk scheduler, il disco, gli interrupt, il kernel, ecc. Elencali nel corretto

[Edit question](#)

ordine temporale, dal momento in cui il processo ha effettuato la system call, al momento in cui riceve i dati.

Paragraph

Path: p

**Question 4**

Not yet answered

Not graded

[Flag question](#)

[Edit question](#)

**Scheduling di processi e kernel**

In un sistema sono presenti tre processi: A, B, C, inizialmente tutti e tre ready nell'ordine A in testa, poi B, C in coda. La politica di scheduling è **round robin** con quanto di tempo pari a 70ms. Attenzione: le system call possono essere bloccanti e i page fault major o minor.

- **A** cpu bound, nessun page fault.
- **B** cpu bound ma genera alternativamente una system call non bloccante e un major page fault ogni 20 ms, ciascun I/O burst è servito in 100ms.
- **C** I/O bound, l'I/O burst dura 100ms.

Il processore esegue di volta in volta A, B, C, e inoltre, con tempi trascurabili, mode switching, dispatching, system call e interrupt handlers. Mostra schematicamente, nella seguente tabella, l'ordine con cui tali attività vengono eseguite (una sola croce per ciascuna colonna). Indica anche quali processi sono running, quali ready e quali bloccati in ciascun istante come indicato nell'esempio.

user mode	A	X	<input type="checkbox"/>									
	B		<input type="checkbox"/>									
	C		<input type="checkbox"/>									
kernel mode	mode switch		<input type="checkbox"/>									
	disptaching		<input type="checkbox"/>									
	system call per I/O		<input type="checkbox"/>									
	interrupt handler per page fault		<input type="checkbox"/>									
stati	interrupt handler per I/O		<input type="checkbox"/>									
	interrupt handler per quanto scaduto		<input type="checkbox"/>									
	running	A	<input type="checkbox"/>									

processi	ready	B							
		C							
	block								
note tempi									
altre note									

**Information**

- Flag question
- Edit question

**Grep e Awk**

Il file di testo [ubuntu\\_packages.txt.gz](#) (devi decomprimerlo usando gunzip) contiene un record per ciascun pacchetto software della distribuzione linux ubuntu. Nel file ciascun record è separato da una linea vuota, i campi sono su linee distinte e ciascuna linea inizia con il nome del campo seguito da ":". Alcuni campi possono mancare. Per svolgere l'esercizio non è necessario conoscere il significato di tutti i campi.

Suggerimenti: alcune volte conviene processare tale file con awk usando RS="" (stringa vuota) e FS="\n".

**Question 5**

- Not yet answered
- Not graded
- Flag question
- Edit question

Nel file ubuntu\_packages.txt le righe che iniziano per "Size:" contengono un numero intero. Dai un comando **bastato su grep o egrep** che selezioni di tali righe quelle il cui contenuto rispetti **entrambe** le seguenti regole:

- il numero sia maggiore di 1 500 000
- l'ultima cifra del numero sia minore o uguale a 4

Paragraph

*Copia e incolla il comando e l'output (almeno le prime righe, usa head se necessario).*

Path: p » em

**Question 6**

- Not yet answered
- Not graded
- Flag question
- Edit question

Considera il campo **Filename**, nel file ubuntu\_packages.txt. Contiene un pathname del file del pacchetto. Tale pathname contiene in quarta posizione il nome di una directory che raggruppa un certo numero di pacchetti. Ad esempio:

```
pool/main/e/empathy/account-plugin-jabber_3.8.4-1ubuntu1_amd64.deb
pool/main/e/empathy/account-plugin-salut_3.8.4-1ubuntu1_amd64.deb
```

**empathy** raggruppa due pacchetti.

Dai una riga di comando che mostri la classifica delle primi dieci di tali directory per numero di pacchetti ospitati. Esempio:

141 mono  
 138 libreoffice  
 105 gcc-4.8  
 89 firefox  
 70 thunderbird  
 ....

Paragraph

*Copia e incolla il comando e il suo output (almeno le prime righe).*

Path: p » em

**Question 7**

Not yet answered

Not graded

Flag question

Edit question

**Debugging**

Considera il codice del seguente progetto [prj3.tar.gz](http://prj3.tar.gz). Compila tutti i file con il comando

```
gcc -g *.c -lm -o fib
```

Considera una esecuzione di **fib** con parametro **21**. Considera la **duecentounesima** volta in cui **fib()** è stata chiamata.

- Mostra lo **stack** in quell'istante.
- Conta quante volte **fib()** è ritornata fino a quell'istante.
- nel contesto di **init\_list()** mostra il penultimo elemento della lista **L**
- Esprimi in una formula la relazione che lega il numero di frame relativi a fib() nello stack, le chiamate a fib(), e i ritorni da fib()

Paragraph

Path: p

**Question 8**

Not yet answered

Not graded

Flag question

Edit question

**Pratica Unix**

Rispondi alle seguenti domande circa la gestione dei processi.

1. Che significa che un processo è in foreground?
2. Che significa che un processo è in background?
3. Descrivi un modo per portare in foreground un processo in background?
4. Descrivi un modo per portare in background un processo in foreground?

Paragraph

- 1.
- 2.
- 3.
- 4.

Path: p

**Question 9**

Not yet answered

Not graded

 Flag question [Edit question](#)**Windows vs. Unix (solo per chi fa 6 cfu)**

Rispondi alle seguenti domande che confrontano Windows e Unix.

1. In Unix gran parte delle informazioni di configurazioni sono in /etc, in Windows dove sono?
2. In Unix gran parte delle system call operano su file descriptor, descrivi il concetto che gioca lo stesso ruolo in Windows.
3. Unix organizza i processi ad albero, in Windows come sono organizzati i processi?

Paragraph

- 1.
- 2.
- 3.

Path: p

[Next](#) [Moodle Docs for this page](#)You are logged in as [Maurizio Pizzonia](#) ([Log out](#))

SOpari20150924