Cognome e nome: _____ Matricola: _____ Posto: ____ ____

Università degli Studi di Padova – Dipartimento di Matematica. - Corso di Laurea in Informatica

Regole dell'esame

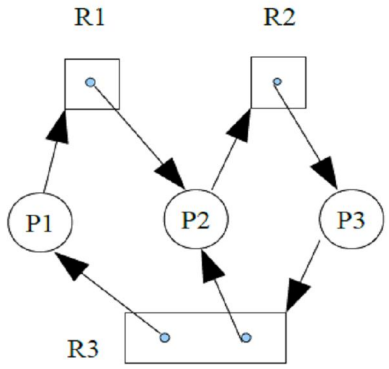
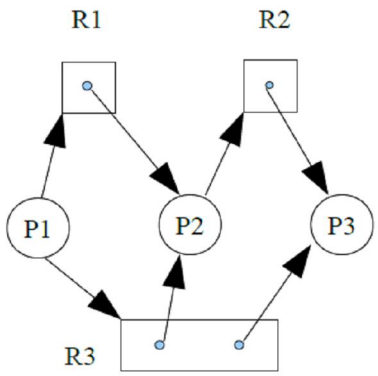
Il presente esame scritto deve essere svolto in forma individuale in un tempo massimo di **75 minuti** dalla sua presentazione.
Non è consentita la consultazione di libri o appunti in forma cartacea o elettronica, né l'uso di palmari e telefoni cellulari.
La correzione, registrazione ed eventuale sessione orale **è stata anticipata al 30 giugno 2014, alle 9.30 in Luf1.**
Per superare l'esame il candidato deve acquisire almeno 1,5 punti nel Quesito 1 e un totale di almeno 18 punti su tutti i quesiti, inserendo le proprie risposte interamente su questi fogli. Riportare generalità e matricola negli spazi indicati.
Per la convalida e registrazione del voto finale il docente si riserva di proporre al singolo candidato una prova orale.

Quesito 0 (1 punto): Scrivere Cognome e Nome in alto in ogni facciata; scrivere la Matricola e il posto sul primo foglio.

Quesito 1 (4 punti): ***0,5 punti per risposta giusta, diminuzione di 0,25 punti per ogni sbaglio, 0 punti per risposta vuota***

DOMANDA	**Vero/Falso**
L'i-node contiene al suo interno i dati dei file (es. il testo scritto da un utente in un file .txt)	
Una system call generata da un processo utente viene gestita in modalità utente	
Un *interrupt* viene gestito in modalità utente	
Il termine *copy-on-write* indica il caso in cui ad ogni modifica di una variabile in memoria RAM si procede immediatamente con l'aggiornamento anche della sua copia su partizione di disco	
In un sistema di memoria a paginazione, il *Translation Lookaside Buffer* (TLB) velocizza la traduzione di indirizzi logici in indirizzi fisici	
Il meccanismo dei semafori consente forme più generali di sincronizzazione tra processi rispetto alla mutua esclusione	
In riferimento allo *scheduling* di processi, la politiche FIFO ed LRU fanno entrambe parte della categoria degli stack algorithms	
Un processo per creare un nuovo processo deve fare una *system call*	

Quesito 2 – (6 punti):
Per ciascuno dei seguenti grafi di allocazione delle risorse, si discuta e commenti se i processi rappresentati si trovino o meno in una situazione di stallo.

Grafo A	**Grafo B**
	

Cognome e nome: _____ Matricola: _____ Posto: ____ ____

Quesito 3 – (6 punti):

Sia data una partizione di disco ampia 256 GB organizzata in blocchi dati di ampiezza 1 KB. Sotto queste ipotesi e assumendo che gli indirizzi debbano essere espressi con un numero di bit multiplo di 8, si determini l'ampiezza massima di file ottenibile per l'architettura di file system **ext2fs** nel caso pessimo di contiguità nulla, assumendo i-node ampi 512 B, i-node principale contenente 13 indici di blocco e 1 indice di I, II e III indirezione ciascuno. Si determini poi il rapporto inflattivo che ne risulta, ossia l'onere proporzionale dovuto alla memorizzazione delle strutture di rappresentazione rispetto a quella dei dati veri e propri.

Effettuati tali calcoli si discuta se e con quale rapporto inflattivo l'architettura **FAT** possa rappresentare file di tale ampiezza nella partizione data, sotto le medesime ipotesi di contiguità nulla.

Cognome e nome: _____ Matricola: _____ Posto: ____ ____

Quesito 4 – (4 punti):

Un sistema di allocazione della memoria ha i seguenti gruppi di pagine libere, in questo ordine:
8KB, 15KB, 3KB, 11KB, 5KB, 13KB, 20KB, 25KB.
Si considerino tre richieste di allocazione che arrivano, una di seguito all'altra, nel seguente ordine:
A) 13KB;
B) 4KB;
C) 11KB.
Indicare a quali pagine vengono assegnate le tre richieste sequenziali A, B e C considerando le politiche *First Fit, Next Fit, Best Fit* e *Worst Fit*. Si assuma che, qualora un gruppo di pagine libere venga assegnato a seguito di una richiesta di dimensione inferiore, il blocco libero sia comunque interamente assegnato.

	A)	B)	C)
First Fit			
Next Fit			
Best Fit			
Worst Fit			

Quesito 5 – (4 punti):

Si consideri la seguente serie di riferimenti a pagine di memoria: 1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6.
Si considerino le seguenti politiche di rimpiazzo:
• LRU
• Optimal
Quanti *page fault* avvengono considerando una RAM con solo 4 *page frame* ed inizialmente vuota?
Si completino inoltre le tabelle mostrando ad ogni istante il contenuto dei 4 page frame di cui è composta la RAM (non è necessario che lo studente mantenga un preciso ordine delle pagine virtuali nelle *page frame*.
Nota: nella tabella la prima riga indica la pagina di memoria virtuale riferita in quell'istante.

Politica di rimpiazzo **LRU**; totale *page fault*? ____

r1	**r2**	**r3**	**r4**	**r2**	**r1**	**r5**	**r6**	**r2**	**r1**	**r2**	**r3**	**r7**	**r6**	**r3**	**r2**	**r1**	**r2**	**r3**	**r6**

Politica di rimpiazzo **Optimal**; totale *page fault*? ____

r1	**r2**	**r3**	**r4**	**r2**	**r1**	**r5**	**r6**	**r2**	**r1**	**r2**	**r3**	**r7**	**r6**	**r3**	**r2**	**r1**	**r2**	**r3**	**r6**

Quesito 6 – (5 punti):

Supponiamo di avere 3 processi che condividono una variabile `x` e che i loro pseudo-codici siano i seguenti:

P1:	**P2:**	**P3:**

`P(SemA)`
`P(Mutex)`
`x=x-2`
`V(Mutex)`
`V(SemC)`
`P(SemA)`
`P(Mutex)`
`x=x-1`
`V(Mutex)`
`V(SemC)`
	`P(SemB)`
	`P(Mutex)`
	`x=x+2`
	`V(Mutex)`
	`V(SemC)`
	`P(SemB)`
		`P(SemC)`
		`P(Mutex)`
		`if (x<0) then V(SemB)`
		`V(Mutex)`
		`P(SemC)`
		`print(x)`

Considerando lo pseudo-codice sopra riportato, determinare se `print(x)` sarà mai eseguita dal processo **P3** e in caso positivo dichiararne l'output. Si assuma che il valore iniziale di `x` sia 1 e che i semafori abbiano i seguenti valori iniziali: `SemA` = 1, `SemB` = 0, `SemC` = 0, `Mutex` = 1.

Cognome e nome: _____ Matricola: _____ Posto: ____ ____

Soluzione

Soluzione al Quesito 1

DOMANDA	Vero/Falso
L'i-node contiene al suo interno i dati dei file (es. il testo scritto da un utente in un file .txt)	F
Una system call generata da un processo utente viene gestita in modalità utente	F
Un *interrupt* viene gestito in modalità utente	F
Il termine *copy-on-write* indica il caso in cui ad ogni modifica di una variabile in memoria RAM si procede immediatamente con l'aggiornamento anche della sua copia su partizione di disco	Più F che V ma entrambe ok
In un sistema di memoria a paginazione, il *Translation Lookaside Buffer* (TLB) velocizza la traduzione di indirizzi virtuali in indirizzi fisici	V
Il meccanismo dei semafori consente forme più generali di sincronizzazione tra processi rispetto alla mutua esclusione	V
In riferimento allo *scheduling* di processi, la politiche FIFO ed LRU fanno entrambe parte della categoria degli stack algorithms	F
Un processo per creare un nuovo processo deve fare una *system call*	V

Soluzione al Quesito 2

Grafo A. Vi sono cicli di processi che hanno assegnate risorse richieste da un altro processo mentre richiedono risorse assegnate a quest'ultimo (attesa circolare).

Ciclo 1: P2 – R2 – P3 – R3 – P2

Ciclo 2: P1 – R1 – P2 – R2 – P3 – R3 – P1

Nessuno dei nodi (processi) del grafo ha solo archi entranti (risorse assegnate); dunque nessuno può completare la propria esecuzione e liberare le risorse in possesso. Il sistema è dunque in stallo. Notare che non è sufficiente indicare solo il ciclo 1 in quanto la risorsa R3 ha molteplicità 2.

Grafo B. Non vi sono cicli e in particolare P3 ha assegnate tutte le risorse richieste; può pertanto completare la propria esecuzione e liberare un'istanza di R3 e R2. Quest'ultima viene assegnata a P2 che può così avere assegnate tutte le risorse di cui necessita per completare la propria esecuzione. Al termine, P2 libera le proprie risorse che possono soddisfare le richieste di P1 permettendogli di completare la propria esecuzione. Il sistema non è in stallo.

Soluzione al Quesito 3

In questa soluzione useremo la notazione informatica tradizionale, con prefissi che denotano potenze di 2.

Essendo la memoria secondaria ampia 256 GB e i blocchi dati ampi 1 KB, è immediato calcolare

che sono necessari: $\left\lceil \frac{256GB}{1KB} \right\rceil$ = 256 M = $2^8 \times 2^{20} = 2^{28}$ indici, la cui rappresentazione binaria banalmente richiede 28 bit.

Stante l'ovvio vincolo che la dimensione dell'indice debba essere un multiplo di un "ottetto" (8 bit), otteniamo la dimensione di 32 bit (4 B).

Sotto queste ipotesi, il file di massima dimensione rappresentabile dall'architettura ext2fs fissata dal quesito sarà composto da:

- 13 blocchi, risultanti dall'utilizzo dei corrispondenti indici diretti presenti nell'i-node principale, al costo di 1 i-node, pari a 512 B
- $\left\lfloor \frac{512B}{4B} \right\rfloor$ = 128 blocchi, risultanti dall'utilizzo dell'intero i-node secondario denotato dall'indice di I indirezione presente nell'i-node principale, al costo di 1 i-node, pari a 512 B
- $128^2 = 2^{14}$ = 16 K blocchi, risultanti dall'utilizzo dell'indice di II indirezione, al costo di 1 + 128 = 129 i-node, pari a: 129 × 512B = (4.096 + 128)B = (2^{16} + 512) B = 66.048 B
- $128^3 = 2^{21}$ = 2 M blocchi, risultanti dall'utilizzo dell'indice di III indirezione, al costo di $1 + 128 + 128^2$ = 16.513 i-node, pari a: 16.513 × 512 B = 8.454.656 B

corrispondenti a 13 + 128 + 16.384 + 2.097.152 = 2.113.677 blocchi ampi 1 KB, al costo complessivo di 1 + 1 + 129 + 16.513 = 16.644 i-node

i-node ampi 128 B, per un rapporto inflattivo di: $\frac{16.644 \times 512\,\text{B}}{2.113.677 \times 1\,\text{KB}} = \frac{16.644}{2.113.677 \times 2}$ = 0,39%.

Vediamo ora di determinare se e in che modo le architetture di file system FAT e NTFS siano in grado di rappresentare file di tale ampiezza sotto le ipotesi fissate dal quesito.

File system di tipo FAT: La struttura FAT, che rappresenta la vista dell'intera partizione in termini di blocchi dati, sarà composta da $\left\lceil \frac{256GB}{1KB} \right\rceil$ = 256 M celle ampie 4 B, una per indice di blocco: di queste, il file che dobbiamo rappresentare ne occuperà 2.113.677, per un rapporto inflattivo — calcolato considerato che l'architettura FAT concettualmente usa l'intera struttura per ogni singolo file — pari a: $\frac{256\,\text{M} \times 4\,\text{B}}{2.113.677\ \text{KB}} = \frac{1\,\text{GB}}{2.113.677\,\text{KB}} = \frac{1.048.576\,\text{KB}}{2.113.677\,\text{KB}}$ = 49,61%.

Nota: anche il caso 28 bit (anziché 32 bit = 4B è stata considerata corretta per quanto discusso in aula sulla FAT-32).

Cognome e nome: _____ Matricola: _____ Posto: ____ ____

Soluzione al Quesito 4

	A)	B)	C)
First Fit	15KB	8KB	11KB
Next Fit	15KB	11KB	13KB
Best Fit	13KB	5KB	11KB
Worst Fit	25KB	20KB	15KB

Soluzione al Quesito 5

Politica di rimpiazzo **LRU**; totale *page fault*? **_10_** (quelli evidenziati)

R1	**r2**	**r3**	**r4**	**r2**	**r1**	**r5**	**r6**	**r2**	**r1**	**r2**	**r3**	**r7**	**r6**	**r3**	**r2**	**r1**	**r2**	**r3**	**r6**
1	**2**	**3**	**4**	2	1	**5**	**6**	2	1	2	**3**	**7**	**6**	3	2	**1**	2	3	6
	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3
		1	2	3	4	2	1	5	6	6	1	2	3	7	6	3	3	1	2
			1	1	3	4	2	1	5	5	6	1	2	2	7	6	6	6	1

Politica di rimpiazzo **Optimal**; totale *page fault*? **_8_** (quelli evidenziati)

R1	**r2**	**r3**	**r4**	**r2**	**r1**	**r5**	**r6**	**r2**	**r1**	**r2**	**r3**	**r7**	**r6**	**r3**	**r2**	**r1**	**r2**	**r3**	**r6**
1	**2**	**3**	**4**	4	4	**5**	**6**	6	6	6	6	**7**	7	7	7	**1**	1	1	1
	1	2	3	3	3	3	3	3	3	3	3	6	6	6	6	6	6	6	6
		1	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3	3	3
			1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2

Soluzione al Quesito 6

Sì, `print(x) = 1`