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 **90 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.
Saranno considerati la chiarezza, il rigore dell'esposizione, la capacità di sintesi, la correttezza e completezza delle risposte.
Riportare con chiarezza qualunque ipotesi aggiuntiva ritenuta necessaria alla risoluzione degli esercizi.
Correzione, registrazione ed eventuale sessione orale: **15 settembre 2014, alle 9:30 in 2BC30.**
Per superare l'esame il candidato deve acquisire 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: Scrivere Cognome e Nome in alto in ogni facciata; scrivere la Matricola e il posto sul primo foglio.

Quesito 1:

Si consideri un sistema che utilizza la paginazione per gestire la memoria.
Si discutano brevemente vantaggi e svantaggi nell'adottare pagine di dimensione ampia oppure di piccola.
[1.A] Pagine di dimensione ampia (vantaggi e svantaggi):

[1.B] Pagine di dimensione piccola (vantaggi e svantaggi):

Quesito 2:

Si consideri la politica di *scheduling Round Robin* di quanto q e si supponga che in un sistema ci siano N processi interattivi tutti con lo stesso comportamento. Ciascuna interazione dà luogo ad un CPU *burst* che richiede la CPU per un tempo c.
[2.A] Se $c < q$ quanto tempo aspetta al più un processo in coda *ready* prima di ottenere la CPU?

[2.B] Se $c < q$ quanto tempo aspetta al più l'utente prima che la CPU finisca di elaborare l'interazione?

[2.C] Se $c > q$ quanto tempo aspetta l'utente prima di iniziare l'ultimo quanto di interazione per l'ultimo processo rimasto ancora attivo? (Si consideri che c può essere espresso come $c = aq + b$; ovvero a quanti di tempo + $b<c$ tempo nell'ultimo quanto)

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

Quesito 3:

E' noto che esistano 4 condizioni che concorrono all'insorgere di situazioni di stallo. Lo studente le elenchi illustrandole brevemente, ovvero in non più di due/tre righe ciascuna.

Quesito 4:

Process A {	Process B {	Process C {
y= y + 2x ;	x= x + 1;	x= y * 3;
}	Print(x); }	Print(y); }

Sincronizzazione di 3 Processi con Semafori.

Per risolvere il seguente esercizio si faccia uso del meccanismo dei semafori.

Si considerino tre processi (A, B e C) i quali devono eseguire alcune operazioni sulle variabili x e y e poi stamparne il risultato finale.

A questo proposito, si consideri la seguente sequenza di avvenimenti:

- I processi A e C devono essere in attesa (ad un semaforo) di essere risvegliati
- Il processo B sveglia il processo A, che a sua volta sveglia il processo C.
- I tre processi accedono concorrentemente (ma in mutua esclusione) alle variabili condivise *x* e *y*, al fine di eseguire le operazioni riportate in figura.
- Nel caso in cui B e C eseguano le loro operazioni su *x* e *y* prima del processo A, i processi B e C si bloccano in attesa che A abbia terminato la computazione su x e y.
- Infine, B stampa il valore di *x* e C stampa il valore di *y* (senza un particolare ordine prestabilito).

Si utilizzino i semafori (es. *SemA*, *SemB*, *SemC* e *Mutex*) **dichiarandone i valori iniziali**.

Assumendo che inizialmente si abbia x = 2 e y = 1, si discutano brevemente (nel retro di questo foglio o del precedente) i possibili valori di x e y al termine dell'esecuzione concorrente di un'istanza dei processi A, B e C.

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

Quesito 5:
Un sistema ha 4 processi e 5 risorse da ripartire. L'attuale allocazione e i bisogni massimi sono i seguenti:

Processo	*Allocate*	*Massimo*
A	1 0 1 1 2	1 1 1 4 2
B	2 0 1 1 1	2 2 2 1 3
C	1 1 1 0 0	2 1 1 0 4
D	1 1 1 0 1	1 1 2 1 3

[5.A] Considerando il vettore delle risorse disponibili uguale a [0 0 1 2 *x*], si discuta per quale valore minimo di *x* questo sia uno stato sicuro e quando invece sia a rischio di deadlock.

[5.B] Per risolvere l'esercizio lo studente ha di fatto ripetutamente utilizzato una parte di un noto algoritmo. Tale algoritmo assegna risorse a processi solo se l'assegnazione fa rimanere il sistema in uno stato sicuro. Come si chiama questo algoritmo?

Quesito 6:
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 ed 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*).

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**

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

Soluzione

Soluzione 1

A) Pagine **ampie**
- Maggiore rischio di **frammentazione interna** ma tabella delle pagine più piccola
  - In media ogni processo lascia inutilizzata metà del suo ultimo *page frame*

B) Pagine **piccole**
- Maggiore ampiezza della tabella delle pagine ma minor frammentazione interna

Altre informazioni possono essere fornite… (si faccia riferimento a slide e libro di testo)

Soluzione 2

[2.A] Un processo aspetta $(N-1)c$ per ottenere la CPU.
[2.B] L'utente aspetta $(N-1)c + c = Nc$
[2.C] L'utente aspetta $N\,qa + (N-1)\,b$

Soluzione 3

Lo studente troverà facilmente la soluzione facendo riferimento alle dispense del corso o al libro di testo.

Soluzione 4

Una versione corretta dei tre processi è quella che segue; con SemA, SemB e SemC inizializzati a 0, e il semaforo Mutex inizializzato a 1.

Process A {	Process B {	Process C {
P(semA);	V(semA);	
V(semC);		P(semC);
P(mutex);	P(mutex);	P(mutex);
y = y + 2*x	x = x + 1	x = y*3;
V(mutex);	V(mutex);	V(mutex);
V(semB);	P(semB);	P(semC);
V(semC);		
	Print(x);	Print(y);
}	}	}

I tre processi potrebbero entrare e uscire dalla sezione critica limitata da P(mutex) e V(mutex) in un ordine qualsiasi dunque per determinare il valore finale di x e y occorre provare tutte le combinazioni.
- Ordine A, B, C: risultato x = 15 e y = 5
- Ordine A, C, B: risultato x = 16 e y = 5
- Ordine B, A, C: risultato x = 21 e y = 7
- Ordine B, C, A: risultato x = 3 e y = 7
- Ordine C, A, B: risultato x = 4 e y = 7
- Ordine C, B, A: risultato x = 4 e y = 9

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

Soluzione 5

[5.A] La matrice delle necessità (massimo numero di risorse richieste dal processo - risorse allocate al processo) è la seguente:

0 1 0 3 0
0 2 1 0 2
1 0 0 0 4
0 0 1 1 2

Se x = 0 oppure x = 1, la deadlock è immediata.
Se x = 2, il processo D può essere eseguito fino alla fine. Quando ha finito, il vettore delle risorse disponibili è [1 1 2 2 3]. Sfortunatamente ora il sistema si trova in deadlock.
Se x = 3, dopo D, il vettore delle risorse disponibili è [1 1 2 2 4] e C può essere eseguito. Dopo il suo completamento, il vettore delle risorse disponibili diventa [2 2 3 2 4]; questo permette a B di essere eseguito e completato. Il vettore delle risorse disponibili diviene dunque [4 2 4 3 5], permettendo il completamento di A.
Quindi il valore più piccolo di *x* per evitare il verificarsi di deadlock è 3.

[5.B] L'Algoritmo del Banchiere (Banker's Algorithm)

Soluzione 6

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

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 in grassetto)

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