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

Università degli Studi di Padova - 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.
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.

Quesito 0 (punti 1): ***Riportare generalità e matricola in alto in tutti i fogli del compito***

Quesito 1 (punti 4): ***1 punto per risposta giusta, diminuzione di 0,33 punti per risposta sbagliata, 0 punti per risposta vuota***

[1.A] Quale tra le seguenti affermazioni concernenti la politica di ordinamento *Round-Robin* è corretta:

1. il tempo di attesa di un processo è sempre maggiore o uguale del suo tempo di risposta
2. il tempo di attesa di un processo è sempre minore o uguale del suo tempo di risposta
3. il tempo di attesa di un processo è sempre uguale al suo tempo di risposta
4. il tempo di attesa di un processo e il suo tempo di risposta non hanno alcun legame prefissato.

[1.B] In quale tra i seguenti sistemi operativi è più conveniente l'utilizzo di *Inverted Page Tables*:

1. nessuno dei seguenti, il vantaggio è pari per tutti
2. sistemi a 16 bit
3. sistemi a 32 bit
4. sistemi a 64 bit

[1.C] Una *system call* bloccante causa sempre un *context switch*:

1. Sempre
2. Mai
3. Sì ma solo se la macchina ha più di un processore
4. Sì ma solo se c'è qualche altro processo attivo

[1.D]: Un semaforo binario può:

1. assumere solo valori discreti
2. gestire solo l'accesso a due risorse condivise
3. gestire solo le richieste di accesso provenienti da due processi
4. assumere solo i valori 0 e 1, con essi denotando "risorsa occupata" e "risorsa libera".

Quesito 2 – (5 punti):
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 2 1 1	1 1 2 1 4
B	2 0 1 1 1	2 2 3 2 1
C	1 1 0 1 0	2 1 3 1 0
D	1 1 1 1 0	1 1 3 2 1

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

[2.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?

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

Quesito 3 – (6 punti):

[3.A] La dimensione massima di un file ottenibile con file system ext2fs dipende dalla contiguità con cui sono scritti i blocchi del file su disco? *Sì / No* ?

[3.B]La dimensione massima di un file ottenibile con file system FAT dipende dalla contiguità con cui sono scritti i blocchi del file su disco? *Sì / No* ?

[3.C] La dimensione massima di un file ottenibile con file system NTFS dipende dalla contiguità con cui sono scritti i blocchi del file su disco? *Sì / No* ?

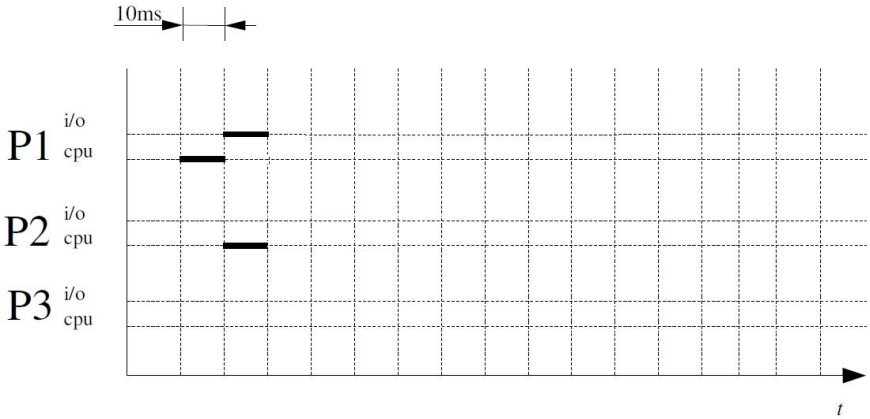
[3.D] Sia data una partizione di disco ampia 64 GB organizzata in blocchi dati di ampiezza 1 KB. In caso serva, si consideri l'ipotesi di contiguità nulla di un file (ciascun blocco si trova su disco in posizione non adiacente al blocco precedente e a quello successivo nella composizione del file).
Si determini l'ampiezza massima di file ottenibile per l'architettura di file system **ext2fs** assumendo i-node ampi esattamente un blocco, i-node principale contenente 12 indici di blocco, 1 indice di I indirezione e 1 indice di II indirezione.
Si determini inoltre la quantità di spazio occupata dalla struttura di i-node necessaria a rappresentare un tale file.

Quesito 4 – (4 punti):

Si considerino tre processi che alternano fasi di uso di una CPU (*job*) con fasi di I/O *burst*. Si utilizzi una politica di CPU *scheduling* del tipo *shortest remaining time next* (SRTN) applicata sui vari job pronti per l'esecuzione sulla CPU.
I tre processi, denominati P1, P2 e P3, sono caratterizzati dalle seguenti sequenze di CPU *burst* e I/O *burst*.

P1: cpu-10ms, i/o-10ms, cpu-30ms, i/o-10ms, cpu-10ms.
P2: cpu-20ms, i/o-10ms, cpu-10ms.
P3: cpu-50ms, i/o-10ms, cpu-10ms.

Ovviamente nessun processo può avanzare alla fase successiva senza aver prima completato le precedenti, nell'ordine.
Si supponga che i tre processi facciano I/O su dispositivi distinti. Si mostri, in ciascun istante di tempo, quali processi risultano in stato di *running* sulla CPU e quali in blocco I/O, marcando, nel diagramma sottostante, le linee di CPU o di I/O corrispondenti a ciascun processo.



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

Quesito 5 – (4 punti):

Si consideri un sistema composto da quattro processi (P1, P2, P3, P4), e quattro tipologie di risorse (R1, R2, R3, R4) con disponibilità: 1 risorsa di tipo R1, 1 risorsa di tipo R2, 1 risorsa di tipo R3, 2 risorse di tipo R4.

Si assuma che:

- ogni volta che un processo richieda una risorsa libera, questa venga assegnata al processo richiedente;
- ogni volta che un processo richieda una risorsa già occupata, il processo richiedente deve attendere che la risorsa si liberi prima di potersene impossessare (utilizzando una coda FIFO di processi in attesa di una determinata risorsa)

Si consideri la seguente successione cronologica di richieste e rilasci di risorse:

1) P2 richiede R1, R2, R3 2) P3 richiede R2, R4	3) P2 rilascia R2 4) P4 richiede R4	5) P1 richiede R1 6) P2 richiede R2	7) P3 richiede R3

Verificare con un grafo di allocazione risorse se alla fine di questa serie di operazioni il sistema si trovi in condizioni di stallo.

Quesito 6 – (6 punti):

Il problema del "produttore/consumatore" è un classico problema di sincronizzazione tra più processi che accedono concorrentemente a risorse condivise. Lo studente utilizzi i **monitor** per scrivere due procedure chiamate `Producer` e `Consumer` che possano essere eseguite concorrentemente al fine di risolvere il problema evitando il *deadlock* del sistema.

(Si consideri il caso in cui le risorse prodotte e non ancora consumate possano essere al massimo `N`).

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

Soluzione

Soluzione al Quesito 1

[1.A]: risposta 1
[1.B]: risposta 4
[1.C]: risposta 4
[1.D]: risposta 4

Soluzione al Quesito 2

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

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

Se x = 0 oppure x = 1, la deadlock è immediata.
Se x = 2, il solo processo D può essere eseguito fino alla sua terminazione. Alla terminazione di D, il vettore delle risorse disponibili è [1 1 3 2 2]. A questo punto può essere eseguito il solo processo C fino alla sua terminazione dopodiché il vettore delle risorse disponibili risulterà essere [2 2 3 3 2]. Segue quindi il processo B e alla sua terminazione il vettore delle risorse disponibili sarà [4 2 4 4 3]. Infine anche A può completare e terminare liberando tutte le risorse.
Quindi il valore più piccolo di *x* per evitare il verificarsi di deadlock è 2.

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

Soluzione al Quesito 3

[3.A] No

[3.B] No

[3.C] Sì

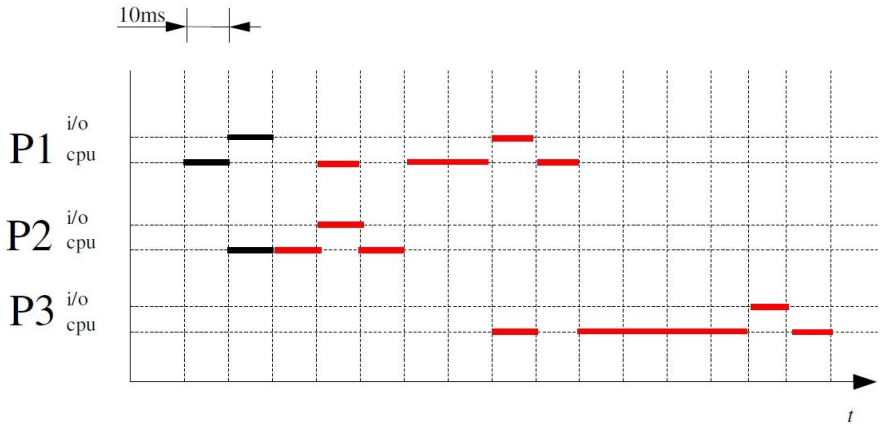
[3.D] In questa soluzione useremo la notazione informatica tradizionale, con prefissi che denotano potenze di 2.
Essendo la memoria secondaria ampia 64 GB e i blocchi dati ampi 1 KB, è immediato calcolare
che sono necessari: $\left\lceil \frac{64GB}{1KB} \right\rceil$ = 64 M = $2^6 \times 2^{20} = 2^{26}$ indici, la cui rappresentazione binaria banalmente richiede 26 *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:

- 12 blocchi, risultanti dall'utilizzo dei corrispondenti indici diretti presenti nell'i-node principale, al costo di 1 i-node, pari a 1 KB
- $\left\lfloor \frac{1024B}{4B} \right\rfloor$ = 256 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 aggiuntivo, pari a 1 KB
- $256^2 = 2^{16}$ blocchi, risultanti dall'utilizzo dell'indice di II indirezione, al costo di 1 + 256 = 257 i-node aggiuntivi, pari a: 257 × 1 KB = 257 KB

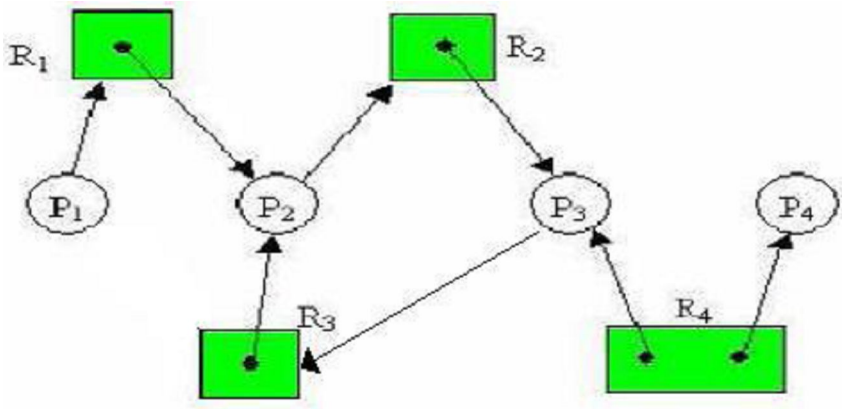
In totale avremo dunque un ammontare di 12 + 256 + 65536 = 65804 blocchi di dati ampi 1 KB, corrispondenti a 65804 KB (dimensione massima dei dati in un file)
al costo complessivo di 1 + 1 + 257 = 259 i-node ampi 1 KB

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

Soluzione al Quesito 4



Soluzione al Quesito 5



Alla fine delle operazioni descritte, il grafo di allocazione delle risorse appare come in figura. Come è evidente, esiste un ciclo di richieste/assegnazioni che coinvolge P2, R2, P3,R3: pertanto, il sistema è in stallo.

Soluzione al Quesito 6

Varie soluzioni possibili, ad esempio:

```
monitor ProducerConsumer
        condition full, empty;
        integer count;
        procedure insert(item: integer);
        begin
                if count = N then wait(full);
                insert_item(item);
                count := count + 1;
                if count = 1 then signal(empty)
        end;
        function remove: integer;
        begin
                if count = 0 then wait(empty);
                remove = remove_item;
                count := count – 1;
                if count = N – 1 then signal(full)
        end;
        count := 0;
end monitor;
```

```
procedure producer;
begin
        while true do
        begin
                item = produce_item;
                ProducerConsumer.insert(item)
        end
end;
procedure consumer;
begin
        while true do
        begin
                item = ProducerConsumer.remove;
                consume_item(item)
        end
end;
```