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 **60 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.
Per superare l'esame il candidato deve acquisire almeno 18 punti su tutti i quesiti, inserendo le proprie risposte interamente su questi fogli. Per la convalida e registrazione del voto finale il docente si riserva di proporre al candidato una prova orale.

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

[1.A]: Sia dato un sistema di memoria con indirizzi virtuali suddivisi in 4 campi: *a*, *b*, *c*, *d*, i primi 3 dei quali siano utilizzati per indirizzare tre livelli gerarchici di tabelle delle pagine e il quarto campo rappresenti l'*offset* entro la pagina selezionata. Indicare dall'ampiezza di quali campi dipende il numero di pagine indirizzate nel sistema:

1. da quella di tutti e quattro i campi
2. da quella del campo *d*
3. da quella del campo *a* e *d*
4. da quelle dei campi *a*, *b*, *c*

[1.B] La dimensione di una FAT dipende da:

1. la quantità di file memorizzati su disco
2. il numero di partizioni virtuali di un disco
3. la contiguità con cui sono scritti i file su disco
4. la dimensione del disco

[1.C]: 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"

[1.D]: Quale tra le seguenti affermazioni è corretta in relazione alla politica di ordinamento processi "FCFS senza valutazione dell'attributo di priorità":

1. il tempo di attesa è sempre maggiore del tempo di risposta
2. il tempo di attesa è sempre minore del tempo di risposta
3. il tempo di attesa è sempre uguale al tempo di risposta
4. il tempo di attesa ed il tempo di risposta non hanno alcun legame prefissato

RISPOSTE AL QUESITO 1: **A** ____ **B** ____ **C** ____ **D** ____

Quesito 2 (4 punti)

In quale tra i seguenti sistemi operativi è più conveniente l'utilizzo di una tabella delle pagine invertita rispetto ad una regolare

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

Si motivi brevemente la risposta (il solo indovinare la risposta, senza motivazione, non fa conseguire alcun punto).

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

Quesito 3 (6 punti):
In una chiavetta USB da 2 GB con filesystem FAT32 e blocchi da 4KB vengono registrati 100 files da 1000 blocchi ciascuno.

Considerando che:
il tempo di lettura di un blocco è 100 μs;
il tempo medio di accesso (tempo per raggiungere qualsiasi blocco da un altro qualsiasi, purché non contiguo altrimenti il tempo è nullo) è di 5 ms;
si presuma che la FAT sia precaricata in memoria (si trascurino pertanto i tempi di accesso alla stessa).

Calcolare il tempo necessario a leggere tutti i files, nelle ipotesi

a) di minima frammentazione;

b) di massima frammentazione;

Si calcoli inoltre
c) il tempo di scanning dello spazio libero, nel caso a)

Quesito 4 – (6 punti):
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.

A) Pagine di dimensione ampia (vantaggi e svantaggi):

B) Pagine di dimensione piccola (vantaggi e svantaggi):

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

Come visto in classe, il valore ottimo di dimensione di una pagina può essere definito matematicamente.
Utilizzando i seguenti parametri:

- σ byte dimensione media di un processo
- π byte dimensione media di una pagina
- ε byte per riga in tabella delle pagine

C) si scriva innanzitutto una funzione f(π) che definisca matematicamente lo spreco di memoria in maniera dipendente dalla dimensione π di una pagina.

- **f(π) =**

D) si determini quindi il valore ottimo di π che minimizza lo spreco di memoria (**mostrare procedimento/calcoli**).

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 se alla fine di questa serie di operazioni il sistema si trovi in condizioni di stallo (suggerimento: usare un grafo di allocazione delle risorse).

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

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 4
[1.B]: risposta 4
[1.C]: risposta 4
[1.D]: risposta 3

Soluzione al Quesito 2

In quello a 64 bit. Infatti, usando tabelle delle pagine regolari, ci si trova ad avere una tabella delle pagine per ogni processo attivo di dimensione proporzionale allo spazio di indirizzamento. Con 64 bit lo spazio di indirizzamento è molto più grande rispetto agli altri casi (2^{64} unità di indirizzamento per ogni processo in confronto a 2^{32} e 2^{16}, considerando rispettivamente 64, 32 e 16 bit per indirizzo). Quindi maggiori sono i bit usati e maggiore sarà l'overhead causato dalle tabelle delle pagine.
Usando invece la tabella delle pagine invertita si ha una sola tabella, indipendentemente dal numero di processi, la cui dimensione è proporzionale alla dimensione della RAM (e.g., indirizzando ogni Byte di una RAM a 4GB si avrebbero 2^{32} unità di indirizzamento a prescindere dal numero di processi). L'overhead causato dalla tabella delle pagine invertita rimane dunque costante (se non varia la RAM) e comunque meno gravoso rispetto ai casi precedente.
E' inoltre evidente che il risparmio in termini di overhead è maggiore quanto maggiore sono i bit usati per ogni indirizzo.

Soluzione al Quesito 3

a) nel caso di minima frammentazione, tutti i blocchi dei files sono contigui e disposti uno di seguito all'altro. Si tratta perciò di leggere 100 x 1000 = 100.000 blocchi;
per ognuno servono 100 microsecondi, dunque il risultato è 100.000 x 100 microsecondi = 10 secondi.
(e' considerato corretto anche il risultato che conta 5 ms iniziali per l'accesso al primo blocco: 10 s + 5 ms)

b) nel caso di massima frammentazione, dato che ci sono in tutto 2^31 / 2^12 blocchi = 524.288 blocchi, è possibile considerare che fra ogni blocco di dati ce ne sia uno libero. Pertanto i files sono tutti "polverizzati". Ad ogni lettura va sommato quindi un "salto" (perché non si tratta di blocchi contigui):
tempo totale = 10 secondi (dal punto precedente) + 100000 x 5 ms = 8 minuti e 30 secondi.

c) i blocchi liberi (escludendo lo spazio per la FAT stessa) sono 524288 - 100000 = 424288, tutti contigui; pertanto si tratta di circa 42 secondi.

Soluzione al Quesito 4

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*

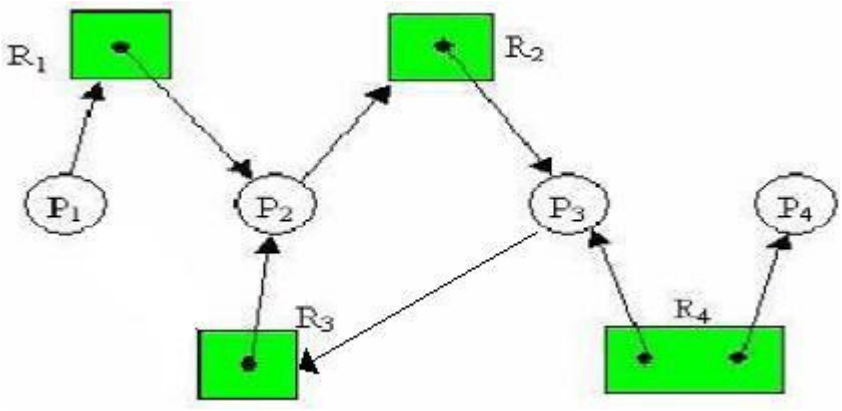
B) Pagine **piccole**
- Maggiore ampiezza della tabella delle pagine ma minor frammentazione interna

C) Spreco per processo come $f(\pi) = (\sigma / \pi) \times \varepsilon + \pi / 2$
- Parte di tabella delle pagine + frammentazione interna

D) Derivata prima è $- \sigma \varepsilon / \pi^2 + ½$
- Ponendo uguale a zero si ha che il minimo di $f(\pi)$ si ha per $\pi = \sqrt{(2 \sigma \varepsilon)}$

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

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

Il problema è chiaramente spiegato nel libro di testo e nei lucidi. 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;
```