PCAD a.a. 2018/19 - Scritto 1 luglio 2019

L'esame è composto da domande a risposta multipla ed esercizi a risposta libera. Se richiesto dal testo o se avete dubbi sulla formulazione di una domanda aggiungete una breve spiegazione per giustificare la risposta. Nella stessa domanda ci può essere più di un'affermazione vera. Occorre raggiungere almeno 15 punti per poter far media con il voto della discussione del progetto.

D1 (1 punto) Quando si utilizza una struttura dati concorrente

1. per ogni accesso alla struttura viene garantita l'assenza di race condition
2. per ogni sequenza di accessi alla struttura dati viene garantita l'assenza di race condition
3. per ogni programma viene garantita l'assenza di race condition
4. per ogni operazione su un suo elemento (es un oggetto in una lista) viene garantita l'assenza di race condition

D2 (1 punto) Nei modelli della concorrenza con interleaving:

1. vengono ammesse solo computazioni che rispettano program order
2. vengono ammesse solo computazioni ottenute eseguendo in blocco tutte le istruzioni di ogni singolo programma in un ordine casuale (es programma 2, programma 1, programma 3)
3. viene sempre garantita la mutua esclusione
4. il numero di context switch è limitato superiormente da una costante

D3 (1 punto) Una Synchronized Collection in Java

1. è equivalente ad una barriera di memoria
2. rende sincronizzata una struttura dati non concorrente
3. implementa il pattern singleton in versione thread-safe
4. è un'implementazione della tecnica di confinamento per oggetto

D4 (1 punto) Un mutex

1. è un semaforo con due possibili valori ed uno stack di thread in attesa
2. è un semaforo con tre possibili valori ed una coda di thread in attesa
3. viene usato per evitare riordinamenti di istruzioni in un thread
4. garantisce starvation-freedom se usato per controllare una risorsa condivisa

D5 (1 punto) Un'operazione atomica

1. è sempre implementata tramite lock
2. non è compatibile con il modello di concorrenza basato su consistenza sequenziale
3. elimina possibili interferenze tra diversi thread fino alla fine della sua esecuzione
4. può utilizzare istruzioni hardware per garantire assenza di race condition

D6 (1 punto) Una barriera di sincronizzazione

1. risolve il problema della sezione critica
2. può essere invocata alla fine di blocchi sincronizzati
3. definisce un punto di sincronizzazione all'interno di codice di singoli thread
4. ha come effetto quello di disabilitare le interruzioni hardware

D7 (2 punti) Il problema della sezione critica

1. si applica a programmi che non condividono memoria
2. si risolve scegliendo in maniera casuale un thread (tra quelli in attesa)
3. si risolve applicando Peterson a due thread scelti casualmente (tra quelli in attesa)
4. è formulato per programmi concorrenti con due o più thread di esecuzione

D8 (2 punti) Quando usiamo oggetti callable in Java

1. Le chiamate dei metodi corrispondenti possono restituire valori
2. Le chiamate dei metodi corrispondenti sono effettuate in mutua esclusione
3. Le chiamate dei metodi corrispondenti sono tutte effettuate in maniera sincrona
4. Non possiamo propagare le eccezioni al di fuori dei metodi corrispondenti

D9 (2 punti) Nella libreria RMI

1. Ogni programma può esportare un solo oggetto remoto
2. La comunicazione è sempre unidirezionale
3. I parametri di un metodo remoto non devono essere necessariamente oggetti serializzabili
4. L'implementazione dell'interfaccia di un parametro deve essere la stessa su server e client

D10 (4 punti) Considerare il seguente codice C:

```
…
int idx=0;
void *foo(void *vargp) {
  idx++;
  printf("Thread %d\n", idx);
  return(0);
}

int main() {
  pthread_t tid[5];
  void *ret;
  for (int i = 0; i < 5; i++) {
     pthread_create(&tid[i], 0, foo, 0);
  }

  for (int i = 0; i < 5; i++) {
     pthread_join(tid[i],&ret);
     idx--;
  }
  return 0;
}
```

1. Il programma può produrre l'output 1 2 3 4 5? (motivare la risposta)

2. Il programma può produrre l'output 2 2 5 5 5? (motivare la risposta)

3. Il programma può produrre l'output 1 1 1 1 1? (motivare la risposta)

4. Il programma può produrre l'output 1 2 1 2 3? (motivare la risposta)

Esercizio 1 (8 punti)
Descrivere una possibile implementazione di una barriera di sincronizzazione tramite semafori e variabili condizione.

Girare foglio per Es. 2

Esercizio 2 (8 punti)
Spiegare a cosa servono i clock logici e quali vantaggi si ottengono passando da clock scalari a vettoriali.