Compito Scritto di Ingegneria del Software

12 settembre 2011

Parte teorica, punti 14

Tempo a disposizione: 1 ora

Si svolga l'esercizio 1 su un foglio e il 2 su un foglio separato

Esercizio 1

Si descrivano i metodi agili per lo sviluppo di software.

PUNTI 7

Esercizio 2

Si descrivano brevemente le principali funzionalità di un IDE avanzato (Netbeans o Eclipse), tra cui refactoring, debugger e profiler.

PUNTI 7

Compito Scritto di Ingegneria del Software

12 settembre 2011

Parte pratica, punti 18

Tempo a disposizione: 1 ora e mezza

Si svolgano gli esercizi 3 e 4 su un foglio e il 5 su un foglio separato

Esercizio 3

Si modelli con una rete di Petri un calolatore multiprocessore dotato di due CPU (CPI 1 e CPU 2) e di un bus che le collega alla memoria. Le due CPU elaborano dati che poi devono scrivere in memoria utilizzando il medesimo bus. All'inizio entrambe le CPU sono in fase di elaborazione dati e il bus è libero. Quando una CPU termina l'elaborazione, deve scrivere il dato prodotto in memoria. Può farlo solo se il bus è libero. In questo caso, il bus diventa occupato dalla CPU fino al termine del trasferimento. Quando si verifica questo evento, il bus torna libero e la CPU riprende la fase di elaborazione.

PUNTI 5

Esercizio 4

Si dia una specifica in Z di un sistema di scheduling dei processi di un sistema operativo batch. Si supponga che il sistema disponga di quattro code a diversa priorità, con la coda 1 a priorità più alta e la coda 4 a priorità più bassa. Si supponga inoltre che le code possano contenere al massimo 10 processi e che i processi siano identificati da un numero intero.

Si modellino in Z le seguenti operazioni:

1) Accodamento di un processo: dato l'identificativo del processo e la sua priorità (un numero da 1 a 4), inserire il processo nella coda relativa alla priorità. L'operazione fallisce se la coda è piena
2) Esecuzione di un processo: si rimuove il processo avente l'identificavo più basso dalla coda avente priorità più alta. L'operazione fallisce se tutte le code sono vuote.

PUNTI 7

Esercizio 5

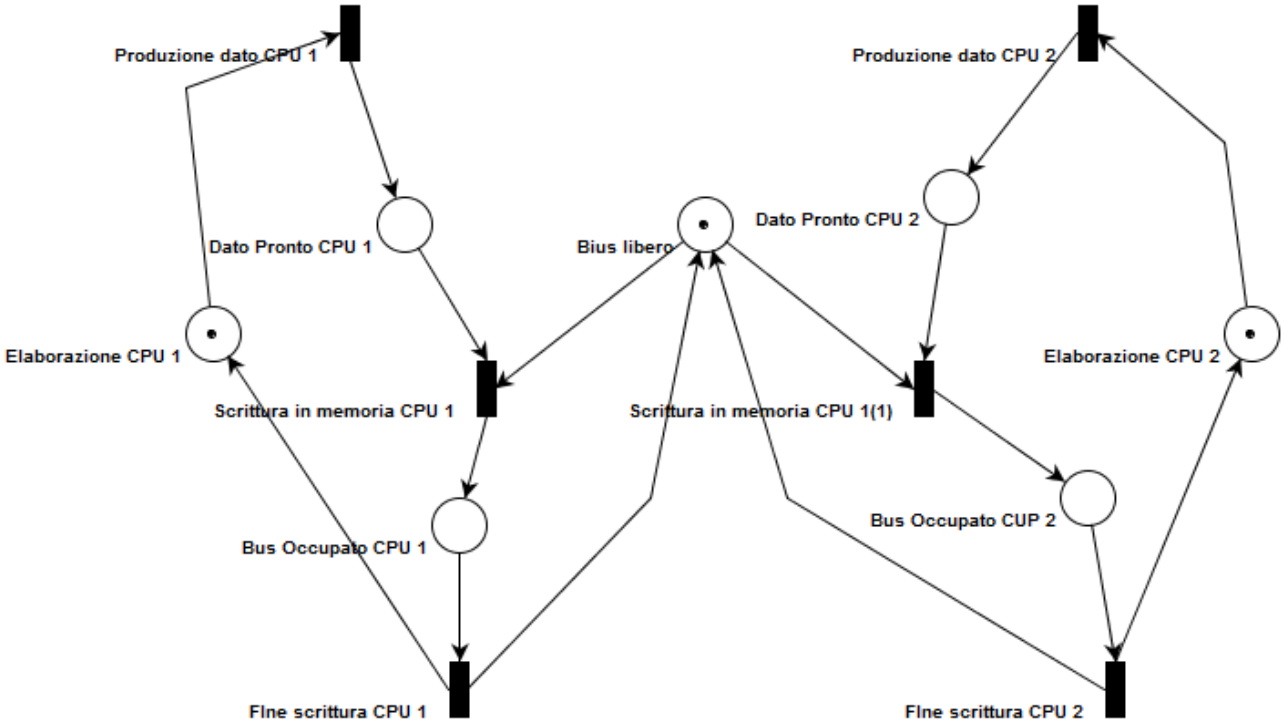
Si disegni il flusso di esecuzione e si esegua l'analisi di flusso, per mezzo di espressioni regolari, per le variabili del seguente programma.
Si individuino inoltre eventuali sequenze non corrette di operazioni e, per ognuna, almeno un caso di test che porti alla sua esecuzione.

```
1. #include<stdio.h>
2. int main (void) {
3.      int a, b, c = 1;
4.      scanf("%d", &a);
5.      while ( a > c ){
6.           b=a-c;
7.           if (a > b) {
8.                a = a - b;
9.                b = b + 1;
10.               c = a + b;
11.                    }
12.          else {
13.               b = a / c;
14.          }
15.     }
16.     c = a - 1;
17.     printf("%d\n",  b );
18.     return 0;
19.     }
```

PUNTI 6

Soluzione

Esercizio 3



Esercizio 4

Tipi definiti dall'utente:

[Code]

Code={1,2,3,4}

Variabili che descrivono lo stato del sistema:

1) coda: è una funzione totale da Code alle sequenze di interi

SO

coda: Code → ℙ ℕ

∀ c: Code • |coda(c)|<=10

InitSO

ΔSO

$coda(1)'=\{\}$
$coda(2)'=\{\}$
$coda(3)'=\{\}$
$coda(4)'=\{\}$

Success

$rep!: Report$

$rep! = 'Okay'$

1) Accodamento:

AccodamentoOK

ΔSO
$processo?: \mathbb{N}$
$pirorità?: Code$

$|code(priorità?)'|<10$
$code''=code \oplus \{priorità? \mapsto code(priorità?) \cup \{processo?\}\}$

CodaPiena

ΞSO
$pirorità?: Code$
$rep!: Report$

$|code(priorità?)'|=10$
$rep! = 'Coda\ piena'$

$$Accodamento \mathrel{\widehat=} AccodamentoOK \wedge Success$$
$$\vee$$
$$CodaPiena$$

2)iEsecuzione

EsecuzioneOK

ΔSO

processo!: $\mathbb{N}$

$\exists$ c: Code $\bullet$ $\forall$ c': Code | c'<c $\bullet$

code(c')= {} $\wedge$ $\exists$ p: $\mathbb{N}$ | p $\in$ code(c) $\bullet$ $\forall$ p': $\mathbb{N}$ | p' $\in$ code(c) $\bullet$ p<p' $\wedge$ processo!=p $\wedge$

code'=code $\oplus$ {c $\mapsto$ code(c) \ {processo?}}

CodeVuote

ΞSO

$\forall$ c: Code $\bullet$ |coda(c)|=0

rep! = 'Code vuote'

Esecuzione $\widehat{=}$ EsecuzioneOK $\wedge$ Success

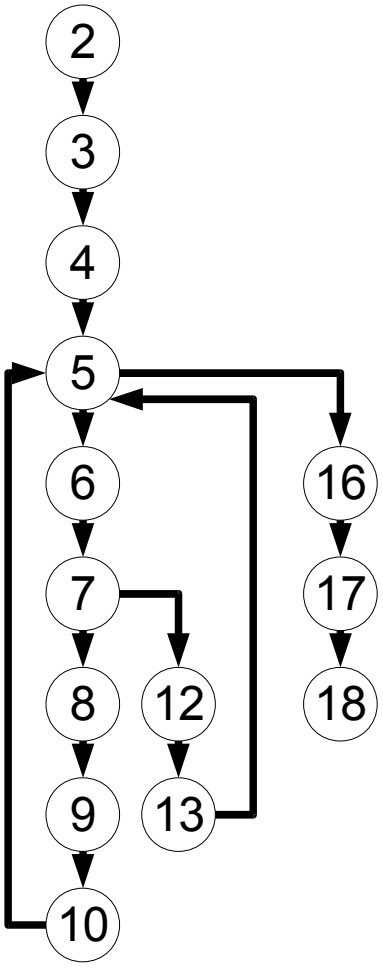
$\vee$

CodeVuote

Esercizio 5

Si disegni il flusso di esecuzione e si esegua l'analisi di flusso, per mezzo di espressioni regolari, per le variabili del seguente programma.
Si individuino inoltre eventuali sequenze non corrette di operazioni e, per ognuna, almeno un caso di test che porti alla sua esecuzione.

```
1. #include<stdio.h>
2. int main (void) {
3.      int a, b, c = 1;
4.      scanf("%d", &a);
5.      while ( a > c ){
6.           b=a-c;
7.           if (a > b) {
8.                a = a - b;
9.                b = b + 1;
10.               c = a + b;
11.                    }
12.          else {
13.               b = a / c;
14.               }
15.          }
16.     c = a - 1;
17.     printf("%d\n",  b );
18.     return 0;
19.     }
```



Riga	a	b	c
3	a	a	ad
4	d		
5	u		u
6	u	d	u
7	u	u	
8	ud	u	
9		ud	
10	u	u	d
11			
12			
13	u	d	u
16	u		d
17		u	
18			
19			

Variabile a: adu (uu(udu + u) u)* u
Variabile b: a (du (uudu + d))* u
Variabile c: adu (u (d + u) u)* d

La variabile b potrebbe essere usata senza essere definita. Questo accade se non si esegue il ciclo while, ovvero per a<=c (quindi per a<=1).
Inoltre, alla riga 16 la variabile c viene definita senza poi essere usata (questo accade sempre).