NOME ____________________ COGNOME __________________________ MATRICOLA_________

Ingegneria del Software II - 21 Gennaio 2014

Parte teoria, punti 14 - Tempo a disposizione: 1h

Esercizio 1 (punti 6)
Si descriva il modello di processo di sviluppo software a cascata.

Esercizio 2 (punti 8)
Si descrivano le specifiche algebriche e se ne fornisca un esempio.

NOME ____________________ COGNOME __________________________ MATRICOLA_________

Ingegneria del Software II - 21 Gennaio 2014

Parte teoria, punti 14 - Tempo a disposizione: 1h

Esercizio 1 (punti 6)
Si descriva il modello di processo di sviluppo software a cascata.

Esercizio 2 (punti 8)
Si descrivano le specifiche algebriche e se ne fornisca un esempio.

NOME ____________________ COGNOME __________________________ MATRICOLA_________

Ingegneria del Software II - 21 Gennaio 2014

Parte teoria, punti 14 - Tempo a disposizione: 1h

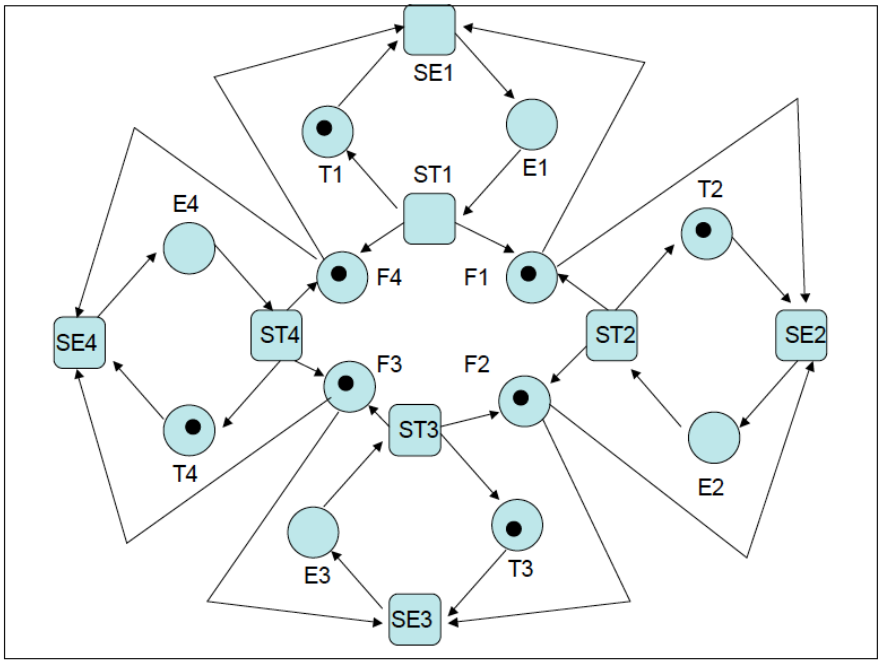
Esercizio 1 (punti 6)
Si descriva il modello di processo di sviluppo software a cascata.

Esercizio 2 (punti 8)
Si descrivano le specifiche algebriche e se ne fornisca un esempio.

NOME ___________________ COGNOME _________________________ MATRICOLA_________

Ingegneria del Software II - 21 Gennaio 2014
Parte pratica, punti 18 - Tempo a disposizione: 2h

Esercizio 3 (punti 5)
La seguente rete di Petri modella il famoso problema dei “Dining philosophers”: ci sono 4 filosofi, a cena, e 4 forchette. Ogni filosofo pensa, oppure mangia utilizzando le due forchette vicine a lui sul tavolo.



Il singolo i-simo filosofo è modellato con due posti:

T_i: Thinking

E_i: Eating

e due transizioni:

SE_i: Start Eating

ST_i: Start Thinking

Per mangiare ogni filosofo ha bisogno delle due forchette adiacenti a lui (modellate con i posti F_i, $F_{[(i-1+4) \text{ modulo } 4]}$).
Inizialmente tutti i filosofi pensano e nessuna forchetta è impugnata.

Scrivere le matrici I e O della rete data e il vettore marcatura iniziale m_0. Si considerino posti e transizioni secondo questo ordine:
Transizioni: SE1, SE2, SE3, SE4, ST1, ST2, ST3, ST4,
Posti: F1, F2, F3, F4, E1, E2, E3, E4, T1, T2, T3, T4
Individuare poi quali transizioni sono abilitate (motivare il perché sulla base della rappresentazione matriciale).
Ci possono essere situazioni di starvation in questa rete? (motivare la risposta)

Esercizio 4 (punti 7)
Un circolo biliardo ha 10 tavoli da biliardo (identificati da 1 a 10).
Per prenotare un tavolo in uno dei giorni della settimana corrente (dove i giorni da lunedì a sabato sono identificati con l’intero da 1 a 6), mattino o pomeriggio, una persona deve occorre specificare il proprio

codice fiscale (CF), il giorno scelto (intero da 1 a 6), la fascia oraria di mattino o pomeriggio (codice char che vale ‘m’ o ‘p’). Se esiste un tavolo libero per giorno e fascia oraria scelti, si assegna il tavolo al CF specificato dalla persona (e è restituito il numero del tavolo assegnato).
Ogni persona può effettuare al massimo una prenotazione in un dato giorno.
Si specifichi in Z un tale sistema di gestione delle sale del cineclub e in particolare la prenotazione di una proiezione.

Esercizio 5 (punti 6)
Si individui un'insieme di casi di test di dimensione <u>minima</u> che soddisfi il criterio di copertura delle decisioni per il programma seguente:

```
program Gennaio2014;
var X, Y, Z, T : integer;
var POS_X, POS_Y : bolean;
begin
        POS_X := true;
        POS_Y := false;
        read(X);
        read(Y);
        if X < 0
                then
                        begin
                        POS_X := false;
                        X := -X
                        end;
        if Y > 0
                then
                        POS_Y := true
                else
                        Y := -Y
        end;
        Z := 0;
        while X <= Y do
                begin
                        X := X - Y;
                        Z := Z + 1;
                end;
        if not (POS_X or POS_Y)
        then T := -X
        else T := X;
        write(Z);
        write(T);
end.
```

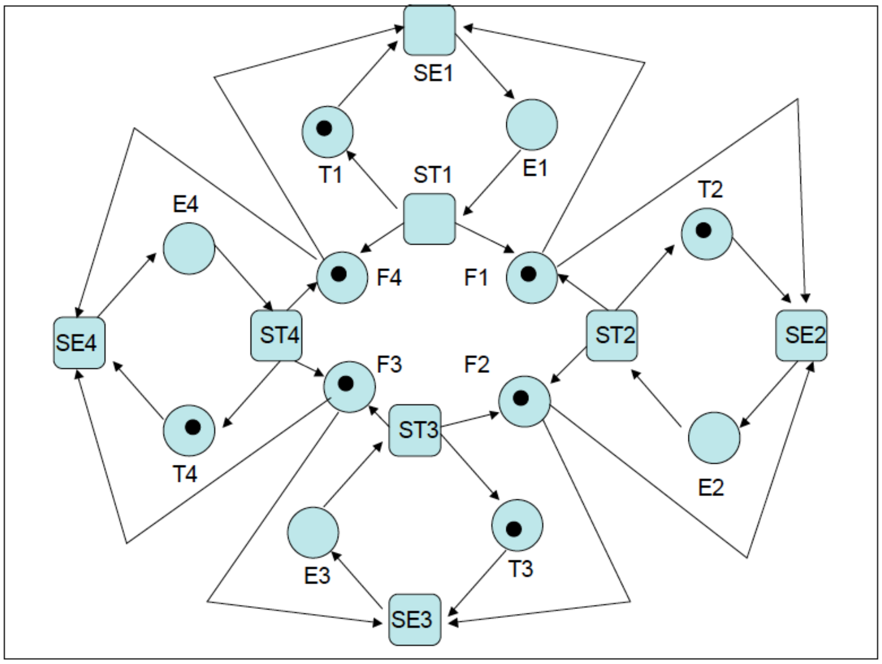
Simulare l’esecuzione della procedura usando i casi di test individuati e commentare i risultati.

NOME ____________________ COGNOME __________________________ MATRICOLA_________

Soluzione

Esercizio 3

Rappresentazione matriciale delle RdP
Matrici I e O: tante colonne quante sono le transizioni (otto nel caso in esame), tante righe quanti sono i posti (dodici in questa rete)



Transizioni: SE1, SE2, SE3, SE4, ST1, ST2, ST3, ST4,
Posti: F1, F2, F3, F4, E1, E2, E3, E4, T1, T2, T3, T4

Matrice I: (ove non indicato nulla si legga 0)

	SE1	*SE2*	*SE3*	*SE4*	*ST1*	*ST2*	*ST3*	*ST4*
F1	1	1						
F2	0	1	1					
F3	0		1	1				
F4	1			1				
E1	0				1			
E2	0					1		
E3	0						1	
E4	0							1
T1	1							
T2	0	1						
T3	0		1					
T4	0			1				

NOME ____________________ COGNOME __________________________ MATRICOLA_________

Matrice O: (ove non indicato nulla si legga 0)

	SE1	SE2	SE3	SE4	ST1	ST2	ST3	ST4
F1					1	1		
F2						1	1	
F3							1	1
F4					1			1
E1								
E2								
E3								
E4								
T1					1			
T2						1		
T3							1	
T4								1

Vettore m0:

F1	1
F2	1
F3	1
F4	1
E1	0
E2	0
E3	0
E4	0
T1	1
T2	1
T3	1
T4	1

Sono abilitate tutte le transizioni SEi (i=1..4).

Ci può essere starvation perché la rete non garantisce che prima o poi ad esempio, una di queste scatti (non è fair come rete).

NOME ____________________ COGNOME __________________________ MATRICOLA_________

Esercizio 4

Tipi definiti dall'utente:

Persone = insieme dei codici fiscali delle persone

Giorni= 1..6

Biliardi = 1..10

Fasce={m,p}

Variabili che descrivono lo stato del sistema:

1. prenotazioni: funzione parziale dai Giorni, Fasce, Biliardi a Persone

Circolo

prenotazioni: Giorni $\times$ Fasce $\times$ Biliardi $\rightarrowtail$ Persone

---

$\forall$ p:Persone, g: Giorni $\bullet$ |{b : Biliardi, f : Fasce| ((g, f, b) $\mapsto$ p) $\in$ prenotazioni }| $\leqslant$ 1

InitCircolo

ΔCircolo

---

prenotazioni'=$\varnothing$

Success

rep!: Report

---

rep! = 'Okay'

NOME ____________________ COGNOME __________________________ MATRICOLA_________

1) Prenotazione

PrenotazioneOK

ΔCircolo
CF?: Persone
giorno?: Giorni
fascia?: Fasce
biliardo!: Biliardi

(giorno?,fascia?, biliardo!) $\notin$dom prenotazioni
$|\{b : Biliardi, f : Fasce| ((giorno?, f, b) \mapsto CF?) \in prenotazioni \}| < 1$
prenotazioni' = prenotazioni $\cup$ $\{(giorno?,fascia?, biliardo!) \mapsto CF? \}$

NessunBiliardoLibero

ΞCircolo
giorno?: Giorni
fascia?: Fasce
rep!: Report

$\nexists$ biliardo: Biliardi • (giorno?,fascia?, biliardo) $\notin$dom prenotazioni
rep! = ‘Nessun biliardo libero’

UnBiliardoOccupato

ΞCircolo
CF?: Persone
giorno?: Giorni
fascia?: Fasce
rep!: Report

$|\{b : Biliardi, f : Fasce| ((giorno?, f, b) \mapsto CF?) \in prenotazioni \}| = 1$
rep! = ‘La persona ha già occupato un biliardo per questo giorno’

Prenotazione$\widehat{=}$ PrenotazioneOK $\wedge$ Success
$\vee$
NessunBiliardoLibero
$\vee$
UnBiliardoOccupato

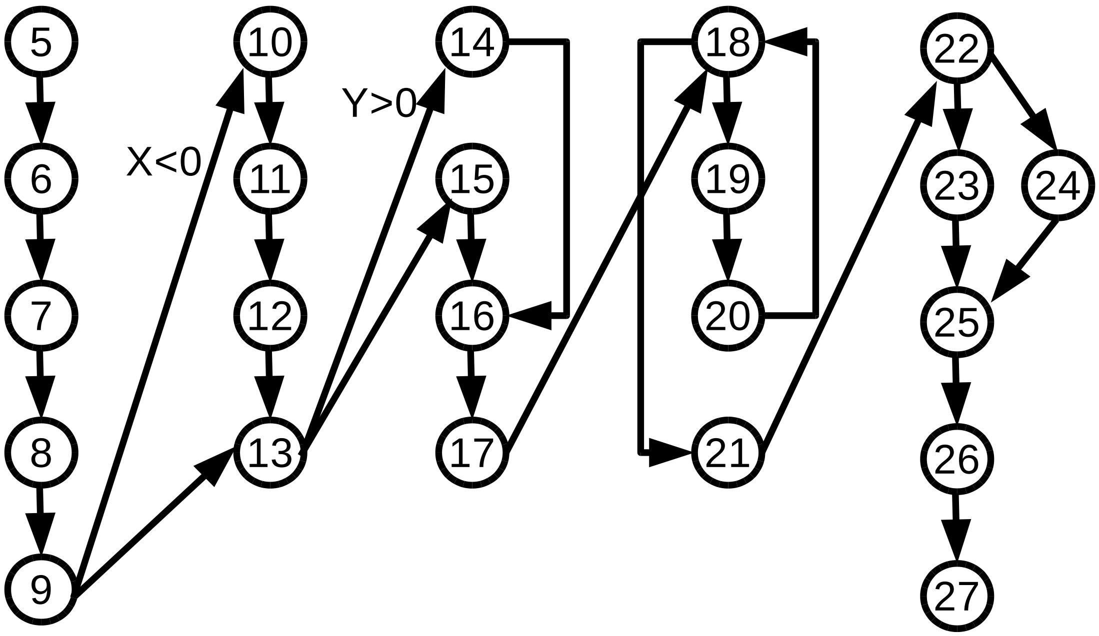
NOME ____________________ COGNOME __________________________ MATRICOLA_________

Esercizio 5

1.	**program** Gennaio2014;
2.	**var** X, Y, Z, T : integer;
3.	**var** POS_X, POS_Y : bolean;
4.	**begin**
5.	POS_X := true;
6.	POS_Y := false;
7.	read(X);
8.	read(Y);
9.	**if** X < 0
	then
	begin
10.	POS_X := false;
11.	X := -X
12.	**end;**
13.	**if** Y > 0
	then
14.	POS_Y := true
	else
15.	Y := -Y
16.	**end;**
17.	Z := 0;
18.	**while** X <= Y **do**
	begin
19.	X := X - Y;
20.	Z := Z + 1;
21.	**end;**
22.	**if not** (POS_X **or** POS_Y)
23.	**then** T := -X
24.	**else** T := X;
25.	write(Z);
26.	write(T);
27.	**end.**

NOME ___________________ COGNOME _________________________ MATRICOLA_________

Il grafo associato risulta:



Per coprire il ramo dal nodo 22 al nodo 23 occorre che la condizione not(POS_X or POS_Y) sia vera, quindi che X e Y siano entrambi negativi.

Caso 1: X<0
Y<0
|X|>|Y|

Con dati di test che soddisfano queste condizioni si coprono tutti gli archi tranne 9-13, 13-15 e 15-16, e gli archi 18-19, 19-20, 20-18 (non si esegue il ciclo while) e 22-24, 24-25

Caso 2: X>=0
Y>=0
X<=Y

Eseguendo il ciclo una volta si esce dal while se:
X-Y > Y ovvero X>2Y

E` una condizione che non può verificarsi (non può essere X<=Y e X>2Y).

Quindi modifichiamo il caso 1, in modo che si percorra il ciclo while almeno una volta:

Caso 1': X<0
Y<0
|X|<=|Y|

Eseguendo il ciclo una volta si esce dal while se:
|X|-|Y| > |Y| ovvero |X|>|2Y|

Il secondo caso diventa:

Caso 2': X>=0
Y>=0
X>Y

(non esegue il ciclo, copre gli archi 9-13, 13-15, 15-16, 22-24 e 24-25).

NOME ____________________ COGNOME __________________________ MATRICOLA_________

I casi 1' e 2' assicurano la copertura di tutti gli archi.
Il test sintetizzato può essere:

T::={(X=-1;Y=-2), (X=2; Y=1)}

L'esecuzione in corrispondenza del primo dato di test corrisponde ad un ciclo infinito (non si esce mai dal while), in quanto all'interno del ciclo il valore di X (diventato +1) viene decrementato di una quantità negativa (-|Y|) e resta pertanto sempre minore di Y (diventato +2).

NOME ______________________ COGNOME _____________________________ MATRICOLA__________