Tema esame inventato
Nuovo - 3

Teoria senza Eclipse:

1. Testing di un programma

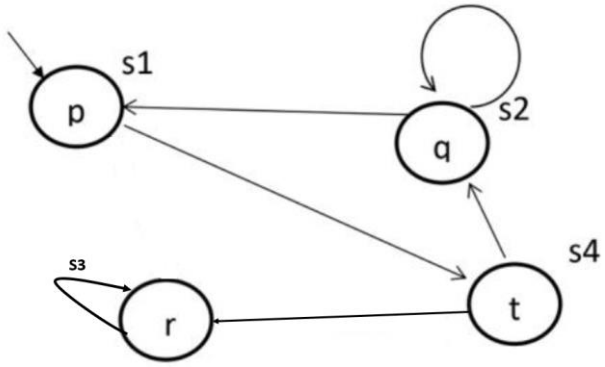
Dato il seguente codice:

```
public class sumCheck{

	// arr: può contenere sia valori positivi che negativi
	// val: è un valore negativo (strettamente)
	int[] arr = {0,0,0,0};
	int val = -1;

	public boolean sum() {
		if(val>=0)
			return false;

		int sum=0;

		for(int x : arr)
			if(x%2!=0 && x<0)
			sum+=x;

		if(sum<=val)
			return true;
		else
			return false;
	}
}
```

Scrivere i casi di test (**simil JUNIT**) per avere la copertura delle **istruzioni**, **branch**, **MCDC** del metodo `sum()` che restituisce true se la somma dei valori negativi e dispari contenuti nell'array *arr* è minore o uguale a *val* (con *val*=numero strettamente negativo).

2. Algoritmo di model checking

Data la seguente macchina M



Mediante l'algoritmo di model checking, dire in quali stati s valgono le proprietà:

- M,s |= AF(p or q)
- M,s |= AG(r)
- M,s |= EX(q or t)

- M,s |= AF(not q)

Nota che proprietà CTL potrebbero aver bisogno di essere trasformate per applicare l'algoritmo di model checking.

3. Combinatorial testing

Date tre variabili con i loro domini

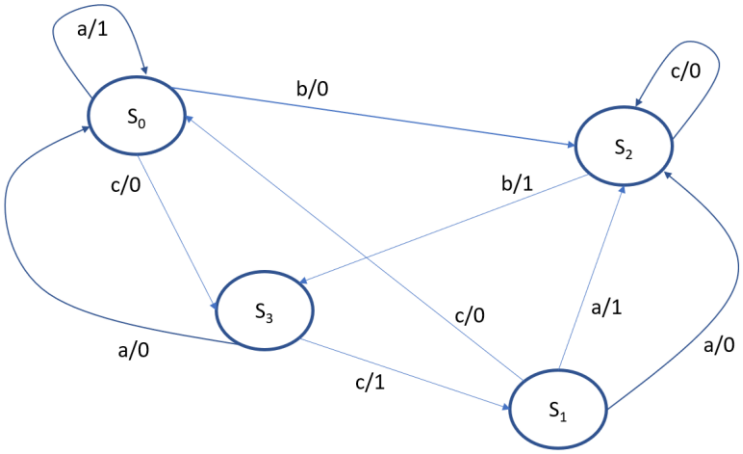
D1: A,B **D2**: 6,7,8 **D3**:L,M,N **D4**:9,2,4

Costruisci la test suite combinatoriale pairwise usando l'algoritmo IPO. Tieni conto dei seguenti **vincoli**:

- A non può andare con L e M
- 6 e L non possono andare insieme con 2

4. Conformance testing

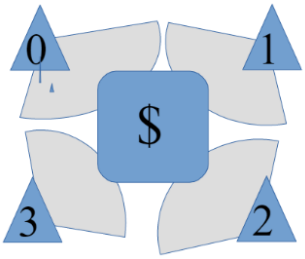
Data la seguente FSM, tre input {a,b,c} e due output {1,0}.



La macchina è corretta? (giustifica la risposta e correggi gli eventuali errori)
Scrivi due test sequences, una per la copertura degli stati e una per la copertura delle transizioni. Fa due esempi di errori (di quelli visti) e controlla se riesci a scoprirli con i test che hai trovato tu.
Può succedere che tu non riesca a scoprire degli errori? Giustifica la risposta.

Con Eclipse:

Dato il seguente problema:



Una cassaforte è protetta sui 4 spigoli da 4 sensori di allarme (numerati da 0 a 3 come in figura). Ogni sensore sorveglia due lati. L'operatore può accenderli e spegnerli, però un sensore può essere spento solo se i suoi due vicini sono accesi (altrimenti si creerebbe un lato non protetto).

All'inizio i sensori sono tutti accesi.

5. Asmeta

Scrivi la specifica Asmeta del problema usando più costrutti possibili (ad esempio derivate, macro rule etc.).

Verificare le seguenti proprietà:

1. non posso mai avere una situazione di pericolo
2. non posso mai avere i sensori 0 e 2 contemporaneamente spenti
3. una volta spento un sensore può essere sempre riacceso
4. un sensore acceso può essere sempre spento
5. se il sensore 0 è acceso e quello 1 è spento, allora 0 rimarrà acceso almeno fino a quando 1 non si accenderà (se si accenderà).

Se c'è qualche proprietà che invece è giustamente falsa e il cui controesempio ti aiuta a capire come funziona, spiegalo. Scrivi almeno una proprietà che è giustamente falsa e controlla che il model checker trovi il contro esempio atteso.

Scrivi uno **scenario Avalla** di un caso significativo (fai un esempio in cui il gioco fa un ciclo completo).

6. JML e KeY

Scrivi il codice in Java del problema implementato prima in Asmeta con in contratti opportuni. Cerca di scrivere sia le precondizioni, che le postcondizioni dei metodi. Cerca di scrivere anche invarianti. Prova i contratti JML con una classe main in cui chiami i diversi metodi. Prova anche a modificare il codice e controlla che i contratti siano violati. Documenta bene le violazioni e le loro cause in commenti.

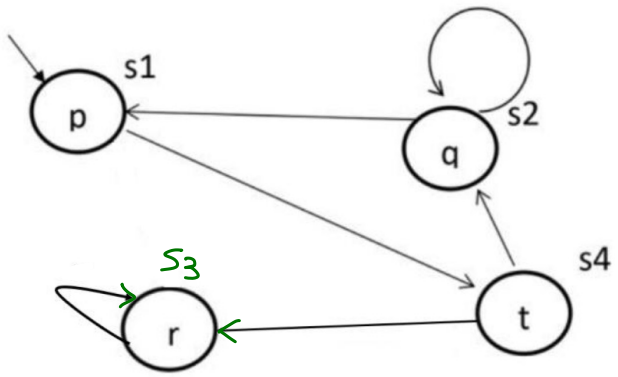
Dimostra con KeY che i contratti siano rispettati (NON USARE ENUMERATIVI).

7. Model based testing

Traduci lo scenario avalla nel **caso di test JUNIT** per il codice Java che hai scritto al punto 6.

2. Algoritmo di model checking

Data la seguente macchina M



1) $AF(p \vee q)$

	S_1	S_2	S_3	S_4
p	x			
q		x		
$p \vee q$	x	x		
$AF(p \vee q)$	x	x		

2) $AG(r) = \neg E(true\ U \neg r)$

	S_1	S_2	S_3	S_4
r			x	
$\neg r$	x	x		x
$E(true\ U \neg r)$	x	x		x
$\neg E(\ldots)$			x	

3) Ex(qVt)

	S_1	S_2	S_3	S_4
q		x		
t				x
qVt		x		x
Ex(qVt)	x	x		x

4) AF(7q)

	S_1	S_2	S_3	S_4
q		x		
7q	x		x	x
AF(7q)	x		x	x

3. Combinatorial testing

Date tre variabili con i loro domini

D1: A,B **D2**: 6,7,8 **D3**:L,M,N **D4**:9,2,4

Costruisci la test suite combinatoriale pairwise usando l'algoritmo IPO. Tieni conto dei seguenti **vincoli**:

- A non può andare con L e M
- 6 e L non possono andare insieme con 2

D1	D2	D3	D4			
A	6	N	9	1°	D1 - D2	✓
A	7	N	2	2°	D1 - D3	✓
A	8	N	4		D2 - D3	✓
B	6	L	4	3°	D1 - D4	✓

B	7	M	9
B	8	N	9
B	6	M	2
B	7	L	4
B	8	L	2
B	8	L	9
B	7	M	4

D2 - D4 V

D3 - D4

AL

AM

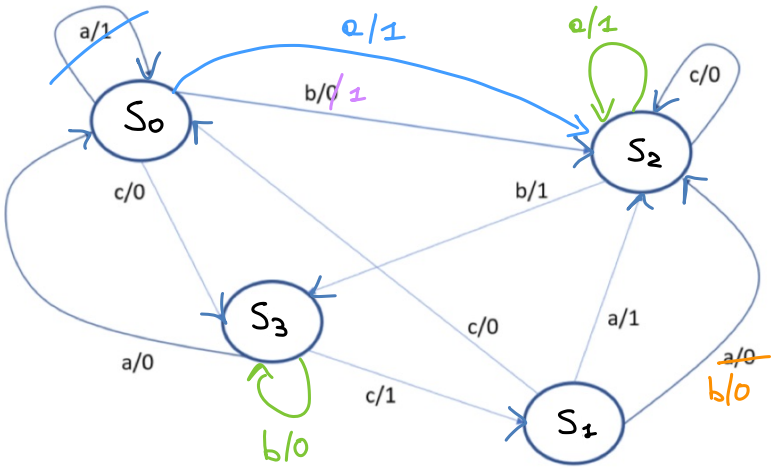
6 L 2

4. Conformance testing

Data la seguente FSM, tre input {a,b,c} e due output {1,0}.

$I = \{a,b,c\}$

$O = \{0,1\}$



La macchina è corretta? (giustifica la risposta e correggi gli eventuali errori)
Scrivi due test sequences, una per la copertura degli stati e una per la copertura delle transizioni. Fa due esempi di errori (di quelli visti) e controlla se riesci a scoprirli con i test che hai trovato tu.
Può succedere che tu non riesca a scoprire degli errori? Giustifica la risposta.

completamente definita deterministica

- Copertura stati : bbcc
  => output atteso : 0110
- Copertura transizioni : bcabbeccebcbbccc = TS2
  => output atteso : 00110001111101101

S_0 : ϕ k ϕ S_1 : ϕ k ϕ S_2 : ϕ k ϕ S_3 : ϕ k ϕ

- Output error : $S_0 \rightarrow b/1 \rightarrow S_2$

  => trovo sia con TS1 che con TS2,

  infatti output : cop. stati = 1110 ≠ TS1

  cop. transizioni = 1.. ≠ TS2

- Transfer error : $S_0 \rightarrow a/1 \rightarrow S_2$

  => non trovo né con TS1 né con TS2, seme status