1. Assegnamento

$$\{p\ [y \to t]\ \}\ y := t\ \{p\}$$

2. Composition rule

$$\frac{\{p\}\ S1\ \{r\},\ \{r\}\ S2\ \{q\}}{\{p\}\ S1;S2\ \{q\}}$$

3a. rafforzare una precondizione

$$\frac{r \to p,\ \{p\}\ S\ \{q\}}{\{r\}\ S\ \{q\}}$$

3b. indebolire una postcondizione

$$\frac{\{p\}\ S\ \{q\},\ q \to r}{\{p\}\ S\ \{r\}}$$

4. If-then-else rule

$$\frac{\{p \wedge e\}\ S1\ \{q\},\ \{p \wedge \neg e\}\ S2\ \{q\}}{\{p\}\ \text{if } e \text{ then } S1 \text{ else } S2 \text{ fi } \{q\}}$$

5. while rule

$$\frac{\{p\} \to \{I\},\ \{I \wedge e\}\ S\ \{I\},\ \{I \wedge \neg e\} \to \{q\}}{\{p\}\ \text{while } e \text{ do } S \text{ od } \{q\}}$$

Il Compitino – linguaggi di programmazione per la sicurezza, prof. Gargantini – 30 /1/2006

Design by contract

1. Cosa sono le **precondizioni**? Chi è responsabile che siano soddisfatte?

2. Esercizio JML:

Si realizzi un metodo statico in Java che dato un array di stringhe restituisce quella più lunga.

- Scrivi le precondizioni e le postcondizioni . Sulle precondizioni sei abbastanza libero. Motiva però le tue scelte. Scrivi le condizioni in sintassi JML (piccoli errori sono ammessi, sempre che si capisca con esattezza la condizione)
- Implementa il corpo del metodo (cioè il codice Java vero e proprio)

3. Xtreme Programming

3. Dato il codice dell'esercizio precedente, scrivi un caso di test (quindi almeno un metodo della classe di test) siginificativo per il metodo che hai implementato nello stile di Junit.
4. Aggiungi un caso di test che chiama il metodo con le precondizioni false: cosa succede in questo caso?
5. Scrivi e spega (in modo sintetico, un paio di righe) **cinque** pratiche proposte da XP che ti hanno colpito.

4. Verifica formale dei programmi

Considera questo programma che prende in input due valori interi x e y > 0 e calcola p = x * y mediante somme ripetute

```
p = 0
q = y
while (q > 0) do
    p = p + x
    q = q - 1
```

Scrivi formalmente la precondizione e la postcondizione e dimostra la correttezza del programma. (Aiuto per trovare l'invariante: ad ogni ciclo p aumenta di x, mentre q parte da y e diminuisce di 1ad ogni ciclo: alla fine p varrà y volte x e q varrà 0).

FSM

Scrivi una macchina a stati finiti che modella un semplice forno a microonde. Il forno

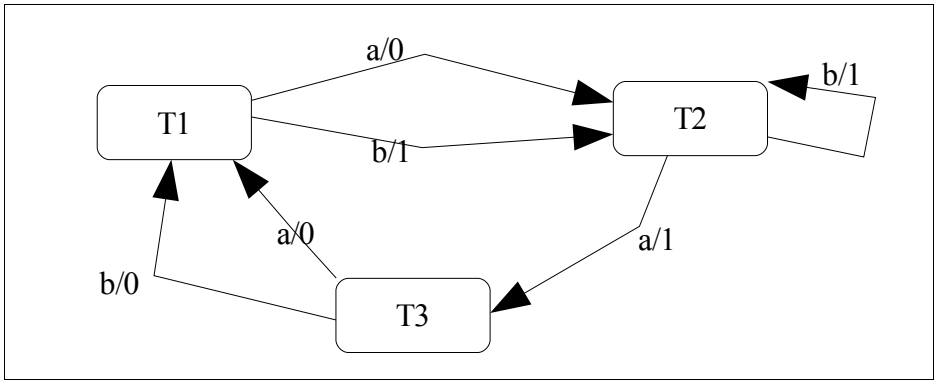
può essere acceso o spento. Quando è acceso ha diverse potenze (ad esempio 360, 500, 700 Watt). All'accensione l'utente seleziona la potenza con un tasto. Si spegne quando uno preme il tasto di off o quando finisce il tempo impostato (non modellare il tempo impostato). Se è acceso e si apre la porta il forno passa in modalità di attesa, e se si richiude la porta riprende a funzionare con la potenza che era stata selezionata. Se la porta è aperta il forno non si accende.

Spiega le scelte che fai (ci sono diverse possibilità, sei abbastanza libero).

Oltre al disegno, indica con esattezza quali sono gli stati, quali gli input che la macchina accetta, e quali eventuali gli output.

Testing per FSM

Data la seguente FSM M, con stato iniziale T1 (senza *status* message). Rispondi alle seguenti domande motivando le risposte.



1) M è minimizzata (ridotta)? (aiuto: applica la definizione di minimizzata ad M)

2) Scrivi una macchina I che implementa M ma introduce un errore di trasferimento in M.

3) La seguente sequenza {a,b,a,a,b,a,b} è un transition tour di M? Scopre l'errore che hai introdotto in I ?

4) In generale un transition tour trova sempre errori di trasferimento?

5) La sequenza {a} è una DS di M?
6) La sequenza {ab} è una DS di M?
7) La sequenza{aab} è una DS di M?

8) Avendo a disposizione un messaggio *reset* che riporta M a T1, scrivi l'insieme delle sequenze date dal metodo di testing basato sulla DS.