

Linguaggi di Programmazione per la Sicurezza

Progettazione di Software Sicuro

(prima parte)

Esame del 24 Gennaio 2013 – (Parte Scritta)

1. Confrontare i linguaggi C e Java in termini di sicurezza. [pt. 3]
2. Classificare l'attività di testing rispetto ai requisiti, all'architettura ed al codice. [pt. 2]
3. Descrivere i modelli di ciclo di vita adatti per lo sviluppo di software sicuro. [pt. 3]
4. Dare la definizione di una FSM estesa e portare un semplice esempio. [pt. 3]
5. Mediante una macchina di stato UML modellare il funzionamento di un *robot* per la pulizia di pavimenti. Il robot è dotato di un *timer* che alle 19:00 attiva il dispositivo. Il robot si muove pulendo il pavimento con dei movimenti ciclici. L'indice di pulizia è dato da un sensore *dirty* che si accende rosso quando il robot inizia a pulire, e diventa verde non appena il pavimento è pulito. Appena il segnale *dirty* diventa verde, il robot si ferma. Il robot è anche dotato di un sensore *tank* che indica la quantità di sporco dentro il serbatoio di accumulo. Tank ha valore verde se il serbatoio non è pieno, altrimenti la spia diventa rossa. Se il serbatoio è pieno, il robot si ferma e riparte solo a serbatoio svuotato. Un errore dei sensori mette il dispositivo in stato di errore. La manutenzione può essere fatta in un qualsiasi momento con interruzione del funzionamento. Quando termina, il dispositivo viene riattivato nella configurazione di interruzione. [pt. 4]
6. Dare la definizione di test set valido, affidabile ed ideale. Dato il seguente programma che dovrebbe restituire il valore della funzione $foo(x, y) = 2(x+y+z)$ per ogni valore x, y interi:

```
int foo (int x, int y, int z) {  
    if (x > 0 & y > x & z > y) {  
        return 2*x+y+z;  
    }  
    else return 2*(x+y+z);  
}
```

fare un esempio di test set ideale. [pt. 3]
7. Scrivere i casi di test secondo l'MCDC per la seguente espressione $(a \& b) \parallel (c \& d)$ [pt.2]