

# Ingegneria del Software II – 01BIF

6 maggio 2011

Non e' possibile consultare libri o appunti. Scrivere le risposte solo sui fogli forniti.  
Sono gradite risposte **brevi** scritte in modo **leggibile**.

Cognome, nome, matricola \_\_\_\_\_

1 (2) Descrivere brevemente la differenza tra verification e validation, e tra correctness e reliability.

Correttezza: proprieta del sistema di funzionare come richiesto per tutti gli ingressi possibili

Affidabilita: probabilita del sistema di funzionare senza failure per un certo tempo

2 (2) Descrivere brevemente la tecnica by analogy, case based per la stima dell'effort di un progetto software

3 (1) Cosa descrive lo standard ISO 12207?

Elenco di attivita in progetti di produzione system e software

4 (1) Quali sono le decisioni fondamentali contenute in un configuration management plan?

Quali elementi del software (documenti o programmi o altro) sono CI, regole per naming, regole per emettere le baseline, regole per versioning, regole e processo per change (serializzazione, CCB etc), tool da usare, responsabile CM

5 (2) Dato un programma diviso in piu' moduli, tipicamente come si distribuiscono i difetti tra loro?

Pochi moduli contengono la maggioranza dei difetti

6 (1) Cosa dice la legge di Weinberg?

Chi produce un documento/programma non e' adatto a verificarlo

7 (2) Si stima per un progetto un effort di 40 mesi persona, con 4-5 persone coinvolte. Quanto potrebbe essere la duration? (dare un range e motivarlo)

Da 8 mesi calendario (se 5 persone lavorano sempre in parallelo)

A 40 mesi calendario (tutte attività sequenziali)

Il limite minimo di 8 mesi è assai improbabile (cfr curva di staffing profile a campana)

Nota: un mese persona si equipara a un mese calendario (con una sola persona che lavora), cioè mese persona = 140 ore persona, cioè circa 35 ore persona a settimana.

8 (2) Occorre sviluppare un progetto di meno di 20 mesi persona, moderatamente mission critical. Che processo software scegliereste e perché?

Per un progetto relativamente piccolo è difficile sostenere un processo rispetto ad un altro, purché in ogni caso si insista sulle attività di V and V.

9 (2) Quali sono vantaggi e svantaggi dell'uso di LOC e Function point per misurare il size di un progetto?

10 (2) Occorre testare un sistema contenente sensori e attuatori, oltre a un programma di controllo su un PC. Come si può procedere?

Occorre separare il test di hw e sw, sostituendo sensori e attuatori con degli stub, e usare questi per il test della parte sw (integrando in modo TD o BU). In seguito si integra hw e sw.

11 (1) Il simple decision coverage implica il decision coverage? Spiegare e/o fare un esempio.

No

Es.  $Y = x \text{ XOR } z$

Test cases	X	Z	Y
T1	T	F	T
T2	F	T	T

T1 e T2 danno simple decision coverage, ma non decision coverage

12 (1) Fare esempi di difetti di integrazione tra moduli

13 (1) Quali sono le fasi del processo software che producono la maggior parte di difetti?

Requisiti e design

14 (1) Come si applica il test risk driven?

Si elencano i rischi peggiori per il software (o sistema) e si cercano i test case che li ottengano. Il test risk driven NON e' il risk management (che si occupa dei rischi di progetto, come sfiorare tempi e costi previsti)

15 (8) Definire i test black box, con tecniche equivalence classes, boundary condition, per la funzione seguente

boolean luggageAllowance(int nLuggagePieces, double totalWeight, int destination);

la funzione calcola se il bagaglio di un passeggero e' accettabile senza sovrapprezzo, con queste regole

-if destination == 0 (Europe), if totalWeight <= 30 return true

-if destination == 1 (Brazil), if nLuggagePieces <=2 return true

-if destination == 2 (US), if nLuggagePieces <=2 and totalWeight <= 30 return true

-else return false

Data la semplicita della funzione e' possibile calcolare in modo dettagliato le combinazioni possibili. Cio evidenzia molti punti indefiniti nella funzione (probabilmente classi non valide, ma comunque da verificare)

Destination	TotalWeight	nLuggagePieces	Classe Valido	Classe Non valida
Minint a -1	Minint a -1	Minint a -1		X
		0 a 2		X
		>2 a maxint		X
	0 a 30	Minint a -1		X
		0 a 2		X
		>2 a maxint		X
	>30 a maxint	Minint a -1		X
		0 a 2		X
		>2 a maxint		X
E cosi via per i valori seguenti				
0				
1				
2				
>2 a maxint				

16 (5) – per la funzione seguente, definire il control flow graph, definire i test case per ottenere il piu alto coverage possibile, secondo i criteri della tabella che segue  
 Per I test case scrivere solo l'input (no valore atteso per l'output)

Coverage type	Feasibility (Y/N)	Coverage obtained (%)	Test cases (riportare nella colonna il nome del test case, scrivere il test case altrove)
Node			
Edge			
Multiple condition (considerare solo riga 4)			
Loop			
Path			

```

1 boolean luggageAllowance(int nLuggagePieces, double totalWeight, int destination){
2     if (destination == 0 && totalWeight == 30) return true;
3     if (destination == 1 && nLuggagePieces <= 2) return true;
4     if (destination == 2 && totalWeight <= 30 && nLuggagePieces <= 2) return true;
5     return false;
}
```