Ingegneria del Software B

Allievi della Laurea Specialistica in Ingegneria Informatica
Tema d'esame - 9 Luglio 2008 – ore 9.00-10.30

NOME: …… COGNOME: …….
MATRICOLA: …….. FIRMA: ……..

Le risposte chiuse valgono 1/30 ciascuna. Il valore degli esercizi è riportato nel prospetto a lato.

Esercizio	**1a**	**1b**	**1c**	**2**	**3**	**4**	**5a**	**5b**	**5c**
Valore	1	2	2	2	2	3	1,5	1,5	3
Valutazione									

Risposte chiuse

Affermazione	**Vera o falsa?**
Nell'esecuzione di un programma orientato agli oggetti che rispetta il principio di sostituzione di Liskov non si possono verificare up-call	
La relazione di dipendenza fra package è transitiva	
Il linguaggio Java supporta l'ereditarietà multipla di un *interface* da più *interface*	
Il design pattern Abstract Factory è classificato come Object Creational (cioè creazionale e basato su oggetti)	
L'attività più costosa nell'ambito della manutenzione è la comprensione del sistema da manutenere	
La squadra che esegue la manutenzione di un sistema è composta interamente da persone che hanno partecipato allo sviluppo del sistema stesso	
Il testing di accettazione è un'attività che fa parte del testing in grande	
L'impalcatura per il testing (stub, driver e oracle) non è necessaria per il testing in grande	
Le ispezioni del codice sono effettuate conclusa l'attività di testing	
La checklist per l'ispezione del codice può comprendere anche controlli di aderenza agli standard aziendali	
Le carte di Gantt servono per risolvere i conflitti di allocazione temporale delle risorse a disponibilità limitata di un progetto	
Ai cinque livelli di maturità di processo stabiliti da CMM (Capability Maturity Model) corrispondono altrettanti livelli di qualità di prodotto	

Esercizi

1) Sia dato il design pattern State.

   a) Classificare tale pattern e descriverne l'intento.
   b) Descrivere la sua struttura.
   c) Derivare un diagramma degli oggetti, contenente almeno due istanze di ogni classe, compatibile con la struttura di cui al punto precedente.

2) Chiarire il concetto di sistema ereditato (*legacy system*).

3) Illustrare graficamente la forza relativa dei criteri di copertura strutturale per il testing del flusso dei dati (*data flow testing*).

4) Creare un insieme di test funzionali di unità per un modulo la cui specifica è la seguente:

   acquisite in ingresso una matrice M quadrata di valori interi, il rango *m* di tale matrice, un valore intero positivo $n \leq m$, determinare se la somma dei valori appartenenti alla diagonale principale della matrice quadrata A di rango *n* contenuta nelle prime *n* righe ed *n* colonne di M assume un valore maggiore di 50.

Si assuma che valga la seguente precondizione: $0 < m \leq 40$.

5) Sia dato il seguente programma.

```
main(){
1     scanf(“%d”, &a);
2     scanf(“%d”, &b);
3     if (a == b)
4          do a--
5          while a > 0;
6     else a = 2;
7     if (b == 3) {
8          a = a + b;
9          b = a*5;
10         if (a == 4) {
11              a--;
12              b = b + 3; }
      }
13    a--;
14    printf(“%d\n”, a);
15    printf(“%d\n”, b);
      }
```

   a) Tracciarne il CFG (Control Flow Graph).
   b) Determinarne l’espressione dei cammini.
   c) Stabilire, mediante esecuzione simbolica, se il cammino <1,2,3,4,5,4,5,7,8,9,10,13,14,15> è percorribile (*feasible*).