Ingegneria del Software B

Allievi della Laurea Specialistica in Ingegneria Informatica
Tema d'esame - 22 Marzo 2004 – 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	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Valore	2	3	2	2	3	3	3
Valutazione							

Risposte chiuse

Affermazione	Vera o falsa?
Nella programmazione a oggetti, l'ereditarietà è una forma di riuso white box	
Il design pattern Observer è classificato come Object Creational (cioè creazionale e basato su oggetti)	
Il design pattern State è classificato come Object Creational (cioè creazionale e basato su oggetti)	
Le ispezioni sono una tecnica di verifica meno efficace del testing	
Le ispezioni devono essere effettuate successivamente rispetto al testing	
L'acquirente di un modulo COTS può sottoporre lo stesso a testing white box	
L'ordinamento dei casi di test in base alla priorità è utile per il testing di regressione	
Le carte di Gantt servono per illustrare l'allocazione temporale dei task in cui un progetto è stato suddiviso	
I metodi di leveling servono per risolvere i conflitti di allocazione temporale delle risorse a disponibilità limitata di un progetto	
ISO 9126 è un modello di qualità di prodotto	
CMM (Capability Maturity Model) è un modello di maturità di processo	
COCOMO II è un modello per la valutazione della qualità di prodotto	

Esercizi

1) Elencare e commentare brevemente alcuni principi di progettazione orientata agli oggetti.

2) Classificare il design pattern Abstract Factory e descriverne sia l'intento, sia la struttura.

3) Puntualizzare la differenza fra testing e debugging.

4) Elencare alcuni criteri di copertura strutturale per il testing di unità.

5) Selezionare un criterio entro l'elenco prodotto al punto precedente e creare un insieme completo di casi di test (test set) che soddisfa tale criterio per il metodo seguente.

```
public void paint(Graphics g){
     if (x1 > 0) {
          int x0, y0, larghezza, altezza;
          if (x1 < x2) {
               x0 = x1;
               larghezza = x2 – x1;
          } else {
               x0 = x2;
               larghezza = x1 – x2;
          }
          if (y1 < y2) {
               y0 = y1;
               altezza = y2 – y1;
          } else {
               y0 = y2;
               altezza = y1 – y2;
          }
          g.drawREct(x0, y0, larghezza, altezza);
     }
}
```

6) Dimostrare, mediante esecuzione simbolica, che il cammino <1,2,3,4,5,6,7,8,11,12,13> del seguente frammento di programma è impercorribile (*infeasible*).

```
main(){
1    scanf("%d", &a);
2    scanf("%d", &b);
3    if (a == 4) {
4         a = a – b;
5         if (a == 3)
6              a--;
     }
7    a--;
8    if (b)
9         while (a)
10             a--;
11   else a = 2;
12   printf("%d\n", a);
13   printf("%d\n", b);
```

7) Sia dato il seguente diagramma Pert dove, a fianco di ogni attività di un progetto, è indicata la durata della stessa, espressa in giorni. Indicare:
   - la finestra temporale relativa a ciascuna attività,
   - il cammino critico,
   - la durata minima del progetto.