Esercizi d'esame per Fondamenti A II - 11/07/2001

Domande a risposta multipla (radiobutton)

1)
Si considerino le interfacce I1, I2 e I3 e la classe A che implementa I3. Valga inoltre:
```
interface I2 extends I1;  interface I3 extends I1, I2;
```
Quale delle seguenti affermazioni è corretta (tenendo conto che si denominano i metodi di una interfaccia i soli metodi specificati dalla interfaccia stessa e non quelli ottenuti per ereditarietà da altre interfacce)?
1. **La classe A deve implementare tutti i metodi della interfaccia I1.**
2. La classe A può implementare solo i metodi della interfaccia I3.
3. La classe A deve implementare solo i metodi della interfaccia I3.
4. La classe non compila perché la relazione tra le interfacce è sbagliata.

2)
Un frammento di codice usa le classi CA, CB (che eredita da CA), CC, e CD (che eredita da CA e da CC). Tutte le classi implementano la interfaccia IntI.
```
CD c; IntI i;
```
Quale affermazione è corretta?
1. **La variabile c non può essere definita, perché la classe relativa è sbagliata.**
2. La variabile i non può essere definita perché la interfaccia non ha senso.
3. Il polimorfismo consentirebbe di assegnare c = i;
4. Il polimorfismo consentirebbe di assegnare CD = CB;

3)
Un frammento di codice usa variabili di classi che implementano le interfacce I1, I2, e I3 (la classi sono C1, C2, e C3, rispettivamente). Le interfacce sono in ereditarietà tra di loro.
```
interface I2 extends I3;
interface I1 extends I2; …
C1 a; C2 b; C3 c; I3 i;
```
1. Non si può avere ereditarietà tra interfacce in disaccordo con gli indici.
2. Si possono fare assegnamenti diretti solo tra le variabili `a` e `b`.
3. **Si possono fare assegnamenti diretti da `a, b, c` alla variabile `i`.**
4. Si possono fare assegnamenti diretti solo dalla variabile `i` alle variabili `b, c`.

4)
Sono state definite in questo ordine:
una interfaccia IA;
una classe A che implementa la interfaccia IA;
una classe SubA astratta che eredita da A e che implementa tutti i metodi della interfaccia IA.
Quale affermazione è vera?
1. Non sono stati rispettati i vincoli temporali di sviluppo.
2. Non è possibile che una classe astratta SubA implementi tutti i metodi di una interfaccia.
3. **La classe SubA ha alcuni metodi astratti.**
4. Si possono creare istanze dalla classe SubA.

5)
Si considerino:
una classe A che definisce i metodi ma, mb e mc;
una sottoclasse B che eredita da A e definisce i metodi ma, mb, mc e md.
una sottoclasse C che eredita da A e definisce i metodi ma, mb, mc e md.
I metodi con lo stesso nome hanno lo stesso numero e tipo di parametri di ingresso e lo stesso tipo di parametro di ritorno.
```
A a; B b; C c;
a = b;  a.ma ();
a = c;  a.ma ();
```
Quale affermazione è vera?
1. Le due invocazioni del metodo ma() sulla variabile `a` sono un esempio di overloading e polimorfismo.
2. **Le due invocazioni del metodo ma() sulla variabile `a` sono un esempio di overriding e polimorfismo.**
3. Le due invocazioni del metodo ma() sulla variabile `a` sono un esempio scorretto.
4. La prima invocazione del metodo ma() sulla variabile `a` è scorretto.

6)
Si consideri una classe A che definisce due costruttori;
Quale affermazione è vera?
1. I due costruttori della classe A sono in overriding e non possono essere disponibili contemporaneamente.
2. **I due costruttori della classe A non possono essere in overriding per potere essere disponibili contemporaneamente**.
3. I costruttori di tutte le sottoclassi devono avere almeno altri due costruttori ciascuna.
4. Non si possono definire due costruttori in una stessa classe a meno che non derivi direttamente da Object.

7)
I controlli per verificare la correttezza delle operazioni di un programma possono essere eseguiti sia durante lo sviluppo di un programma sia durante la esecuzione.
Quale delle seguenti operazioni è falsa?
1. La fase di controllo statico verifica completamente la correttezza di un programma e evita completamente i controlli durante la esecuzione.
2. La fase di controllo dinamico durante la esecuzione è l'unica fase di reale verifica della correttezza di un programma che evita completamente i controlli durante la compilazione.
3. In Java non si eseguono controlli perché è un linguaggio sicuro.
4. **Solo la presenza di controlli sia dinamici sia statici consente di garantire la correttezza di una applicazione generica.**

8)
Quale affermazione è corretta, in relazione alla visibilità di un metodo metPack() di visibilità package?
Per sottoclassi dirette si intendano solo le classi che ereditano esplicitamente da una classe; per sottoclassi si intendano tutte le possibili sottoclassi che ereditano in modo diretto, ed anche in modo indiretto.
1. Il metodo metPack() può essere invocato solo nella classe di definizione e dalle sottoclassi dirette.
2. Il metodo metPack() può essere invocato solo nella classe di definizione e da tutte le sottoclassi.

3. **Il metodo metPack() può essere invocato solo nella classe di definizione e da tutte la classi dello stesso package.**
4. Il metodo metPack() può essere invocato solo nella classe di definizione e da tutte le classi che fanno riferimento al package o ai package in ereditarietà con il package che la contiene.

9)

Un frammento di codice usa le classi CA, CB (che eredita da CA), e CC (che eredita da CB). Tutte le classi implementano la interfaccia IntI.

```
IntI i; CA a; CB b; CC c;
```

Quale affermazione è vera?

1. È corretto assegnare: `b = i;`
2. **È corretto assegnare: `a = b;`**
3. È corretto assegnare: `i = a + b;`
4. È corretto assegnare: `c = i;`

10)

Si consideri il seguente metodo:

```
static public void bubbleSort (int v[], int n){
boolean ordinato = false;
 while (n>1 && !ordinato){
   ordinato = true;
   for (int i=0; i<n-1; i++)
     if (v[i] > v[i+1]) {
     scambia(v,i,i+1);
     ordinato = false; }
     n--;
 }
}
```

1. Il metodo è statico perché non può essere cambiato.
2. Il metodo è statico si vuole limitare la visibilità nello stesso package.
3. **Il metodo è statico perché è definito a livello di classe.**
4. Il metodo è statico perché la variabile ordinato non è resa visbile all'esterno.

11)

Si consideri il seguente metodo:

```
static public void bubbleSort (int v[], int n){
boolean ordinato = false;
 while (n>1 && !ordinato){
   ordinato = true;
   for (int i=0; i<n-1; i++)
     if (v[i] > v[i+1]) {
     scambia(v,i,i+1);
     ordinato = false; }
     n--;
 }
}
```

Quali valori delle variabili e dei parametri producono la uscita dal ciclo while del metodo?

1. Il ciclo non termina mai.
2. Il ciclo termina per n negativo e con ordinato a true.
3. **Il ciclo termina per n negativo o nullo o con ordinato a true.**
4. Il ciclo termina per n negativo o nullo e con ordinato a true.

12)

In relazione alla ricorsione in un metodo generico di una classe generica pubblica:

1. Si può usare la ricorsione solo nei metodi pubblici.
2. Si può usare la ricorsione solo nei metodi non protected nè private.
3. **Si può sempre usare la ricorsione.**
4. Si può usare la ricorsione solo nell'ambito dello stesso package.

13)

Cosa descrive meglio la visualizzazione grafica e la gestione di una interfaccia grafica in Java?

1. Java mette a disposizioni classi sia per oggetti awt sia per oggetti swing.
2. **Java permette di scegliere oggetti awt e oggetti swing che possono essere liberamente mescolati.**
3. Java permette di scegliere oggetti awt e oggetti swing, che non possono essere liberamente mescolati.
4. Java mette a disposizioni classi awt e swing, ma non istanze degli stessi.

14)

Si considerino le linee di codice:

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
```

1. Le clausole di importazione sono in un ordine sbagliato.
2. **Le clausole di importazione sono corrette in qualunque ordine.**
3. La seconda e la terza clausola di importazione devono essere scambiate.
4. La terza clausola di importazione non è necessaria in quanto implicata nella prima.

15)

Se in un metodo compaiono le seguenti istruzioni:

```
if (x == 0) throw new NullPointerException();
```

quale delle seguenti affermazioni è falsa?

1. Il metodo deve prevedere una clausola throws NullPointerException.
2. Il metodo può restituire o l'oggetto del tipo previsto o una eccezione.
3. **Il metodo restituisce sempre un oggetto del tipo previsto.**
4. Il programma intero potrebbe terminare a causa delle istruzioni riportate sopra.