Esonero del corso di Programmazione a Oggetti

Roma, 9 novembre 2006

Considerate le seguenti definizioni di classi e interfacce in Java:

```
class P { public static void print(String s) {System.out.println(s);} }

interface Studente{ void sostieneEsame(int voto);
                void sostieneTesi();}

class Pupa{
   public void bacia(Studente s){P.print("se mi piace!");}
   public void bacia(Secchione s){P.print("giammai!");}
   public void bacia(Goliarda g){ P.print("subito!");}

   public void canta(){P.print("Boys, boys, boys...");}
   public void canta(Studente s) {((Goliarda) s).canta(); }
}

class PupaSecchiona extends Pupa{
    public void bacia(Studente s){
      if (s instanceof StudenteIngegneria) P.print("Fossi matta");
         else if (s instanceof Goliarda) P.print("Non sei serio");
         else P.print("Si puo' fare");}
}

class StudenteUniversitario implements Studente{
   int esamiFatti;
   protected boolean laureato;
   protected int esamiDaFare = 30;
   public StudenteUniversitario(){ esamiFatti=0; laureato=false;}
   public void sostieneEsame(int voto){
        if (!rifiuta(voto))
           { esamiFatti++;
             if (voto>27) P.print("Ho preso "+voto);
                     else P.print("Mi hanno dato "+voto);
            }
```

```
            else P.print("rifiuto!");
       }
 public void sostieneTesi(){if (esamiFatti==esamiDaFare) laureato=true;}
 protected boolean rifiuta(int voto) {return voto<24;}
}

class StudenteIngegneria extends StudenteUniversitario{
   public StudenteIngegneria(){esamiDaFare=50;}
}

class Secchione extends StudenteIngegneria{
   protected boolean rifiuta(int voto){return voto<30;}
}

class Goliarda extends StudenteUniversitario{
 public void sostieneEsame(int voto){
       super.sostieneEsame(voto);
       if (voto>18) P.print("Sono un genio! Beviamoci su!");
}
 protected boolean rifiuta(int voto) {return !super.rifiuta(voto);}
 public void canta(){ P.print("Dottore, dottore...");}
}
```

In tutte le domande, le istruzioni si supporranno scritte in un metodo `main` dove sono state date le seguenti dichiarazioni e creazioni di oggetti:

```
public class TestaStudenti_Pupe{
    public static void main(String args[]){
           Pupa p = new Pupa();
           Pupa ps = new PupaSecchiona();
           StudenteUniversitario sun=new StudenteUniversitario();
           StudenteIngegneria ing = new StudenteIngegneria();
           Secchione sec=new Secchione();
           Goliarda g=new Goliarda();
           Studente s_sun=new StudenteUniversitario();
           Studente s_sec=new Secchione();
           Studente s_ing=new StudenteIngegneria();
           Studente s_g = new Goliarda();
}
```

Domanda 1 L'istruzione `p.bacia(sec)` stamperà
☐ `giammai!`
☐ `se mi piace!`
☐ `subito!`

Domanda 2 L'istruzione `p.bacia(g)` stamperà
☐ `giammai!`
☐ `se mi piace!`
☐ `subito!`

Domanda 3 L'istruzione `p.bacia(ing)` stamperà
☐ `giammai!`
☐ `se mi piace!`
☐ Niente, perché non ci sono versioni del metodo `bacia` con parametro di tipo `StudenteIngegneria`

Domanda 4 L'istruzione `p.bacia((Studente) sec)` stamperà
☐ `giammai!`
☐ `se mi piace!`
☐ `subito!`

Domanda 5 L'istruzione `((Studente) p).bacia(sec)` causerà
☐ Un errore in fase di compilazione, perché `Studente` non ha relazioni di sottotipaggio con `Pupa`
☐ Un errore in fase di esecuzione perché il metodo `bacia()` non è definito nella classe `Studente`
☐ Stamperà `se mi piace!`

Domanda 6 L'istruzione `p.bacia((Goliarda) s_sun)` causerà:
☐ Un errore in fase di compilazione, perché `Goliarda` non ha relazioni di sottotipaggio con `Studente`
☐ Un errore in fase di esecuzione perché `sec` non punta ad un oggetto di tipo `Goliarda`
☐ Stamperà `subito!`

Domanda 7 L'istruzione `p.bacia((Goliarda) sec)` causerà:
☐ Un errore in fase di compilazione, perché `Goliarda` non ha relazioni di sottotipaggio con `Secchione`
☐ Un errore in fase di esecuzione perché `sec` non punta ad un oggetto istanziato dalla classe `Goliarda`
☐ Stamperà `subito!`

Domanda 8 L'istruzione `ps.bacia(sec)` stamperà
☐ `giammai!`
☐ `fossi matta`
☐ `si puo' fare`

Domanda 9 L'istruzione `ps.bacia(s_sec)` stamperà
☐ `giammai!`
☐ `fossi matta`
☐ `si puo' fare`

Domanda 10 L'istruzione `ps.bacia(ing)` stamperà
☐ `se mi piace!`
☐ `fossi matta`
☐ `si puo' fare`

Domanda 11 L'istruzione `ps.bacia(sun)` stamperà
☐ `se mi piace!`
☐ `fossi matta`
☐ `si puo' fare`

Domanda 12 L'istruzione `ps.bacia(s_g)` stamperà
☐ `non sei serio`
☐ `fossi matta`
☐ `subito!`

Domanda 13 L'istruzione `ps.bacia(g)` stamperà
☐ `non sei serio`
☐ `fossi matta`
☐ `subito!`

Domanda 14 L'istruzione `ps.bacia((Goliarda) sun)` causerà:
☐ Un errore in fase di esecuzione perché `sun` non punta ad un oggetto istanziato dalla classe `Goliarda`
☐ Un errore in fase di compilazione, perché `Goliarda` non ha relazioni di sottotipaggio con `StudenteUniversitario`
☐ Stamperà `subito!`

Domanda 15 Dire quali delle seguenti affermazioni è falsa:
☐ `Secchione` è sottotipo di `Studente`
☐ `Secchione` è sottotipo di `StudenteIngegneria`
☐ `Goliarda` è sottotipo di `Secchione`

Domanda 16 L'istruzione `s_g.canta()` stamperà:
☐ `Dottore, dottore...`
☐ Niente, perché darà un errore a tempo di compilazione
☐ Niente, perché darà un errore a tempo di esecuzione
☐ Niente, ma il programma compila ed esegue correttamente

Domanda 17 L'istruzione `((Goliarda)s_g).canta()` stamperà:
☐ `Dottore, dottore...`
☐ Niente, perché darà un errore a tempo di compilazione
☐ Niente, perché darà un errore a tempo di esecuzione
☐ Niente, ma il programma compila ed esegue correttamente

Domanda 18 L'istruzione `g.canta()` stamperà:
☐ `Dottore, dottore...`
☐ Niente, perché darà un errore a tempo di compilazione
☐ Niente, perché darà un errore a tempo di esecuzione
☐ Niente, ma il programma compila ed esegue correttamente

Domanda 19 L'istruzione `if (sec istanceof Goliarda) ((Goliarda) sec).canta()` stamperà:
☐ `Dottore, dottore...`
☐ Niente, perché darà un errore a tempo di compilazione
☐ Niente, perché darà un errore a tempo di esecuzione
☐ Niente, ma il programma compila ed esegue correttamente

Domanda 20 L'istruzione `if (s_sec istanceof Goliarda) ((Goliarda) s_sec).canta();` stamperà:
☐ `Dottore, dottore...`
☐ Niente, perché darà un errore a tempo di compilazione
☐ Niente, perché darà un errore a tempo di esecuzione
☐ Niente, ma il programma compila ed esegue correttamente

Domanda 21 L'istruzione `if (g.laureato) P.print("congratulazioni");` stamperà:
☐ `congratulazioni`
☐ Niente, perché darà un errore a tempo di compilazione
☐ Niente, perché darà un errore a tempo di esecuzione
☐ Niente, ma il programma compila ed esegue correttamente

Domanda 22 L'istruzione `p.canta();` stamperà:
☐ `Dottore, dottore...`
☐ `Boys, boys, boys...`
☐ Niente, perché darà un errore a tempo di esecuzione
☐ Niente, perché darà un errore a tempo di compilazione

Domanda 23 L'istruzione `p.canta(g);` stamperà:
☐ `Dottore, dottore...`
☐ `Boys, boys, boys...`
☐ Niente, perché darà un errore a tempo di esecuzione
☐ Niente, perché darà un errore a tempo di compilazione

Domanda 24 L'istruzione `p.canta(ps);` stamperà:
☐ `Dottore, dottore...`
☐ `Boys, boys, boys...`
☐ Niente, perché darà un errore a tempo di esecuzione
☐ Niente, perché darà un errore a tempo di compilazione

Domanda 25 L'istruzione `p.canta(sun);` stamperà:
☐ `Dottore, dottore...`
☐ `Boys, boys, boys...`
☐ Niente, perché darà un errore a tempo di esecuzione
☐ Niente, perché darà un errore a tempo di compilazione

Domanda 26 L'istruzione `g.sostieneEsame(26);` stamperà:
☐ `rifiuto!`
☐ `rifiuto!`
`Sono un genio!  Beviamoci su!`
☐ `Mi hanno dato 26`

Domanda 27 L'istruzione `s_g.sostieneEsame(23);` stamperà:
☐ `rifiuto!`
☐ `rifiuto!`
`Sono un genio!  Beviamoci su!`
☐ `Mi hanno dato 23`
`Sono un genio!  Beviamoci su!`
☐ Niente, perché darà un errore a tempo di compilazione

Domanda 28 L'istruzione `s_sec.sostieneEsame(28);` stamperà:
☐ `rifiuto!`
☐ `Mi hanno dato 28`
☐ Niente, perché darà un errore a tempo di compilazione