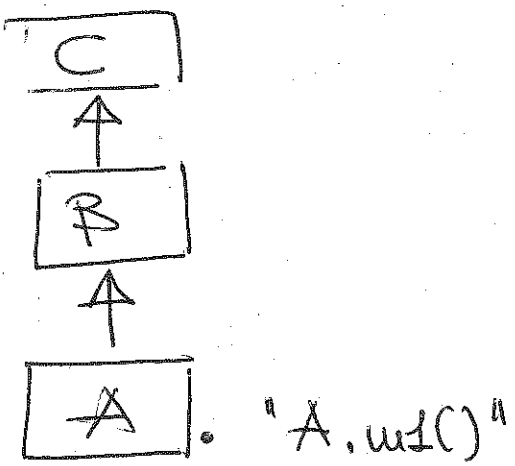
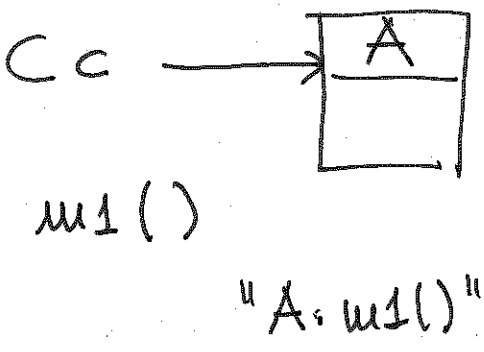
Quiz: Costrutti OO 1

1) Date le seguenti classi:




```
public class C {
public void m1() {
System.out.println("C.m1()");
}
}
public class B extends C{
public void m1() {
System.out.println("B.m1()");
}
}
public class A extends B{
public void m1() {
System.out.println("A.m1()");
}
}
public class P
{
        public static void test(C c){
          c.m1();
        }
}
public class MainClass{
public static void main(String args[]){
        C c=new A();
        P.test(c);
        }
}
```

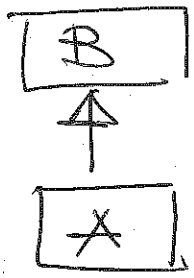


Qual è il risultato della compilazione ed esecuzione del programma ?

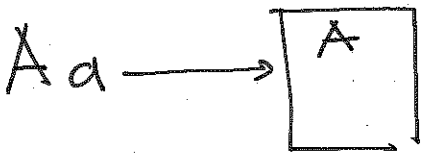
1. Errore di compilazione
2. Viene stampato "A.m1()"
3. Viene stampato "B.m1()"
4. Viene stampato "C.m1()"

Quiz: Costrutti OO 1	Corso: "Programmazione orientata agli oggetti"	Docente: Andrea Bei

2) Date le seguenti classi:




```
public class B
{
private int z;
        public B(int m){
           z=2*m;
        }
        public B(float m){
           z=3*(int) m;
        }
        public int getZ() {return z;}
}
public class A extends B
{
private int x;
private int y;
        public A(int i,int j){
           super((float)i/4);
           x=i;
           y=j;
        }
        public A(int i){
           this(i,10);
        }
        public void print(){
           System.out.println(x+y+getZ());
        }
}
public class MainClass{
public static void main(String args[]){
                A a=new A(5);
                a.print();
        }
}
```



Cosa viene stampato ?:

5. 17
6. 18
7. 15
8. 0

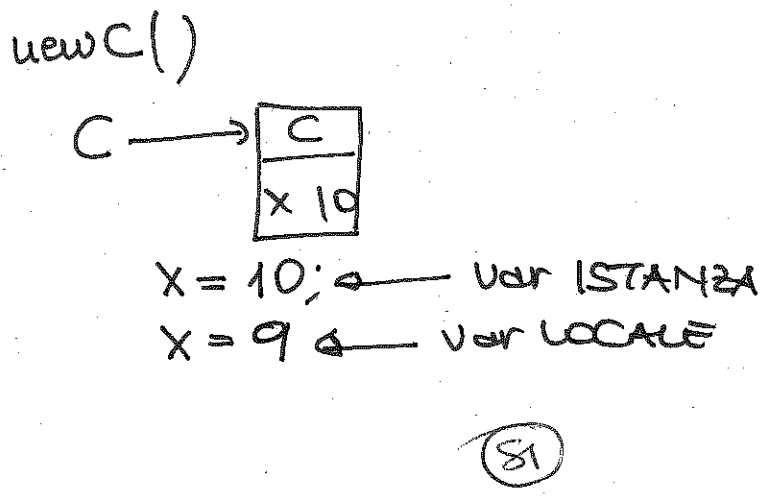
this (5,10)
super((float) 5/4)
z = 3
x = 5 y = 10
3+5+10 → 18

Quiz: Costrutti OO 1	Corso: "Programmazione orientata agli oggetti"	Docente: Andrea Bei

3) Date le seguenti classi:

```
public class C {
private int x;
public C(){
x=10;
}
public void m1(int h) {
System.out.print("x="+x+" ");
  {
  int x=3*h;
  System.out.println("x="+x);
 }
}
}
public class MainClass{
public static void main(String args[]){
      (new C()).m1(3);

          }
}
```

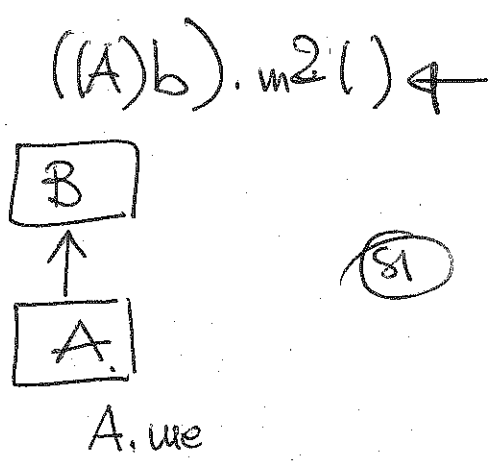


Qual è il risultato della compilazione ed esecuzione del programma ?

1. Errore di compilazione
2. stampa x=10 x=9
3. stampa x=10 x=10
4. stampa x=0 x=9

4)Date le seguenti classi:

```
public class B{
public void m1(){
System.out.println("B.m1()");
}
}
public class A extends B {
          public void m2(){
             System.out.println("A.m2()");
          }
}
public class MainClass{
public static void main(String args[]){
      B b=new A();
      b.m2();
          }
}
```



Qual è il risultato della compilazione ed esecuzione del programma ?

1. stampa: "A.m2()"

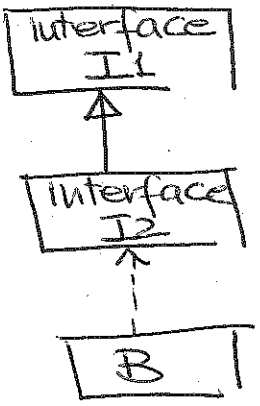
2. stampa: "B.m1()"
3. errore di compilazione
4. errore a tempo di esecuzione

5) Date le seguenti classi:

```
public abstract class B{
public abstract void m1();
public abstract void m2();
}
public class A extends B{
        public void m2(){
           System.out.println("A.m2()");
        }
}
public class MainClass{
public static void main(String args[]){
     B b=new A();
     b.m2();
}
}
```

Qual è il risultato della compilazione ed esecuzione del programma ?

1. stampa: "A.m2()"
2. stampa: ""
3. errore a tempo di esecuzione
4. errore di compilazione



S1

Non implementa m1()
ERRORE a tempo di compilazione

6) Date le seguenti classi:

```
public interface Int1 {
        public void m1();
}
public interface Int2 extends Int1{
        public void m2();
}
public class B implements Int2{
        public void m2(){
        System.out.println("B.m2()");
        }
}
public class MainClass{
public static void main(String args[]){
     B b=new B();
     b.m2();
        }
}
```

Qual è il risultato della compilazione ed esecuzione del programma ?

1. stampa: "B.m2()"
2. stampa: ""
3. Errore di compilazione
4. errore a tempo di esecuzione

7) Date le seguenti classi:

```
public class B
{
private int x;
public B(){
x=10;
}
public int getX(){
        return x;
        }
}
public class A extends B
{
        public void m2(){
         System.out.println("x= "+x);
        }
}
public class MainClass{
public static void main(String args[]){
    (new A()).m2();
        }
}
```

Qual è il risultato della compilazione ed esecuzione del programma ?

1. Errore di compilazione
2. stampa x=10
3. stampa x=0

Quiz: Costrutti OO 1	Corso: "Programmazione orientata agli oggetti"	Docente: Andrea Bei

8) Date le seguenti classi:

```
public class B
{
<??????> int x;
public B(){
x=10;
}
public int getX(){
        return x;
        }
}
public class A extends B
{
        public void m2(){
        x=0;
        }
}
public class C
{
        public void m2(){
        B b=new B() ;
        b.x=2;
        }
}
```

pubblico o PROTETTO ?

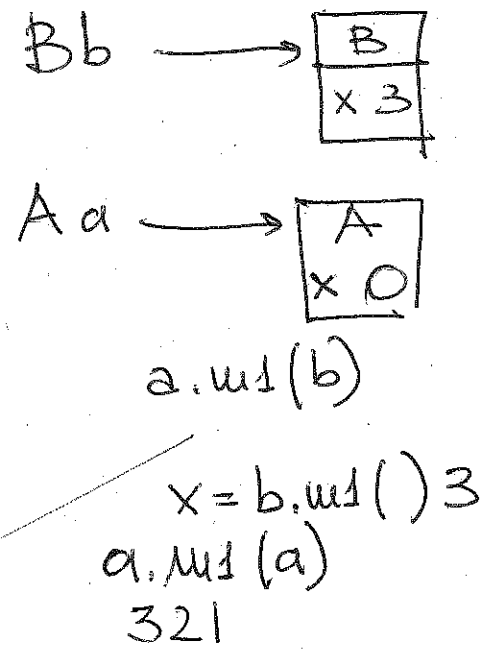
Come deve essere dichiarato l'attributo x di B affinchè la compilazione della classe A vada a buon fine e la compilazione della classe C dia errore ?

1. protected
2. public
3. private

Quiz: Costrutti OO 1	Corso: "Programmazione orientata agli oggetti"	Docente: Andrea Bei

9) Date le seguenti classi:

```
public class B {
int x;
public B(int i){
x=i;
}
public int m1(){
return x;
}
}
public class A
{
int x;
public void m1(B b) {
x=b.m1();
m1(this);
}
public void m1(A a) {
System.out.print(x--);
if (x>0) a.m1(this);
}
}
public class MainClass{
public static void main(String args[]){
        B b=new B(3);
                A a=new A();
                a.m1(b);
        }
}
```



Cosa viene stampato ?:

1. nulla
2. 3
3. 1
4. 321

10) Date le seguenti classi:

```
public class A{
private B b;
public A(B b){
this.b=b;
}
public void m1() {
b=new B();
}
public int getValue(){
return b.getValue();
}
}
public class B{
private int y ;
        public B(){
                y=10;
        }
        public int getValue(){
                return y;
        }
        public void setValue(int j){
                y=j;
        }
}
public class MainClass{
public static void main(String args[]){
        B b=new B();
        A a=new A(b);
        b.setValue(15);
        System.out.print(a.getValue());
        a.m1();
        System.out.print(a.getValue());
        }
}
```

Cosa viene stampato ?

1. nulla
2. 1010
3. 1015
4. 1510
5. 1515

Quiz: Costrutti OO 1	Corso: "Programmazione orientata agli oggetti"	Docente: Andrea Bei

11) Data la seguente classe

```
1:  public class Q
2:  {
3:     int maxElements;
4:
5:     void Q()
6:     {
7:        maxElements = 100;
8:        System.out.println(maxElements);
9:     }
10:
11:    Q(int i)
12:    {
13:       maxElements = i;
14:       System.out.println(maxElements);
15:    }
16:
17:    public static void main(String[] args)
18:    {
19:       Q a = new Q();
20:       Q b = new Q(999);
21:    }
22: }
```

Qual è il risultato della compilazione ed esecuzione del programma ?

1. Stampa 100 e 999.
2. Stampa 999 e 100.
3. Errore di compilazione a linea 3 (variabile maxElements non inizializzata).
4. Errore di compilazione a linea 19.

12) Date le seguenti classi:

```
1  class A
2  {
3    public int r=10;
4    void callme()
5    {
6      System.out.println("A");
7    }
8  }
9  class B extends A {
10
11   public void callme()
12   {
13     System.out.println("B");
14   }
15
16 }
17 class Q
18 {
19   public static void main(String args[])
20   {
21     B a = new B();
22     a.callme();
23     System.out.println(b.r);
24   }
25 }
```

Qual è il risultato della compilazione ed esecuzione del programma ?

1. Errore di compilazione
2. Stampa B e 10
3. Stampa A e 10

Quiz: Costrutti OO 1	Corso: "Programmazione orientata agli oggetti"	Docente: Andrea Bei

13) Date le seguenti classi:

```
1  class Messaggio {
2  String text;
3  public Messaggio() { text ="Hello1"; }
4  }
5  class Super {
6   Messaggio msg;
7     public Super() { msg=new Messaggio(); }
8  }
9  class Ered extends Super
10 {
11    public static void main(String arg[])
12    {
13       Ered i=new Ered();
14       i.print();
15    }
16    public void print()
17    {
18       //Inserire il codice QUI !
19    }
20 }
```

Quale dei seguenti è il modo più semplice di stampare il valore della variabile text a linea 18 ?

1. System.out.println(msg.text);
2. System.out.println(super.msg.text);
3. System.out.println(Messaggio.text);
4. System.out.println(text);

14) Quali delle seguenti dichiarazioni di classe è corretta ?

```
1. public class Fred {
     public int x = 0;
     public Fred (int x) {
       this.x = x;
     }
   }

2. public class fred
     public int x = 0;
     public fred (int x) {
       this.x = x;
     }
   }

3. public class Fred extends MiaClasseBase, MiaAltraClasseBase{
     public int x = 0;
     public Fred (int xval) {
```

non può ereditare da 2 classi

```
    x = xval;
  }
}
```

15) Date le seguenti classi:

```
1.  class Veicolo {
2.  public void guida() {
3.  System.out.println("Veicolo: guida");
4.  }
5.  }
6.  class Automobile extends Veicolo {
7.  public void guida() {
8.  System.out.println("Automobile: guida");
9.  }
10. }
11. public class Test {
12. public static void main (String args []) {
13. Veicolo v;
14. Automobile c;
15. v = new Veicolo();
16. c = new Automobile();
17. v.guida();
18. c.guida();
19. v = c;
20. v.guida();
21. }
22. }
```

Quali delle seguenti affermazioni è vera ?

1. Errore di compilazione su v= c;
2. Errore a tempo si esecuzione su v= c;
3. Stampa:

   Veicolo: guida
   Automobile: guida
   Automobile: guida

5. Stampa:

   Veicolo: guida
   Automobile: guida
   Veicolo: guida

16) Dove, in un costruttore, deve essere inserita l'istruzione super per chiamare il costruttore della superclasse ?

1. Ovunque
2. Deve essere la prima istruzione del costruttore

3. Deve essere l'ultima istruzione del costruttore
4. L'istruzione super non può essere inserita nel costruttore

17) Da quale istruzione nel codice seguente l'oggetto Impiegato("Roberto",48) può essere eliminato dal garbage collector ?

```
1. public class Test {
2. public static void main (String args []) {
3. Impiegato e = new Impiegato("Roberto", 48);
4. e.calcolaPaga();
5. System.out.println(e.stampaDettagli());
6. e = null;
7. e = new Impiegato("Federica", 36);
8. e.calcolaPaga();
9. System.out.println(e.stampaDettagli());
10. }
11. }
```

qui

1. Linea 10
2. Linea 11
3. Linea 6
4. Linea 8
5. Mai

18) Data la seguente classe:

```
public class A {int i1; void m1() {}}
```

Quali delle seguenti affermazioni è vera ?

1. La classe A eredita dalla classe Object. ×
2. Il compilatore inserisce implicitamente un costruttore di default. ×
3. Il costruttore di default accetta un parametro per ogni attributo di A. no
4. Il costruttore di default invoca il costruttore della superclasse × (Object)

19) Date le seguenti classi:

```
class A {A(int i) {}}  // 1
class B extends A {}   // 2
```

B() { super(); }

Quali delle seguenti affermazioni sono vere ?

1. Il compilatore crea un costruttore di default per la classe A no
2. Il compilatore crea un costruttore di default per la classe B sì ma dà errore a tempo di COMPILAZIONE
3. Errore di compilazione a linea 1. no
4. Errore di compilazione a linea 2. sì

20) Quali delle seguenti affermazioni è vera ?

1. Il compilatore crea un costruttore di default solo se non esiste già un costruttore
2. Il costruttore di default ha zero argomenti.
3. Se la classe A ha una superclasse allora il costruttore di default di A invoca il costruttore a zero argomenti della superclasse.

21) Dato il seguente codice :

```
class Q {
 public static void main (String[] args) {
   private int x = 1;
   System.out.println(x);
}}
```

Quali delle seguenti affermazioni sono vere ?

1. Stampa: 1
2. Errore di esecuzione
3. Errore di compilazione
4. Nessuna

22) Data la seguente classe:

```
class Rosso {
 public int a;
 public static int b;
 public static void main (String[] in) {
   Rosso r1 = new Rosso(), r2 = new Rosso();
   r1.a++; r1.b++;
   System.out.print(r1.a+", "+r1.b+", "+r2.a+", "+r2.b);
}}
```

Qual è il risultato della compilazione ed esecuzione del programma ?

1. Stampa: 0, 0, 0, 0
2. Stampa: 0, 1, 1, 1
3. Stampa: 1, 1, 1, 0
4. Stampa: 1, 1, 0, 1
5. Stampa: 1, 1, 0, 0
6. Stampa: 1, 1, 1, 1
7. Compile-time error
8. Run-time error
9. None of the above

23) Quali delle seguenti affermazioni sono vere ?

1. Un metodo final non può essere sovrascritto
2. Tutti i metodi dichiarati in una classe final sono implicitamente final
3. Tutti i metodi dichiarati in una classe final devono essere esplicitamente dichiarati final altrimenti avviene un errore di compilazione

24) Data la seguente classe:

```
class Q {
  static int m1(int x) {return ++x ;}
  public static void main (String[] args) {
    int x = 1;
    int y = m1(x);
    System.out.println(x + "," + y);
}}
```

Qual è il risultato della compilazione ed esecuzione del programma ?

1. Stampa: 1,1
2. Stampa: 1,2
3. Stampa: 2,1
4. Stampa: 2,2
5. Errore di esecuzione
6. Errore di compilazione
7. Nessuna delle precedenti

25) Data la seguente classe:

```
class Q {
  private static int x=1;
  static void m1(int i) {x++; i++;}
  public static void main (String[] args) {
    int y=3; m1(y);
    System.out.println(x + "," + y);
}}
```

Qual è il risultato della compilazione ed esecuzione del programma ?

1. Stampa: 1,3
2. Stampa: 2,3
3. Stampa: 1,4
4. Stampa: 2,4
5. Errore di esecuzione
6. Errore di compilazione
7. Nessuna delle precedenti

26) Data la seguente classe:

```
class Q {
  private String name;
  public Q(String name) {this.name = name;}
  public void setName(String name) {this.name = name;}
  public String getName() {return name;}
  public static void m1(Q r1, Q r2) {
    r1.setName("Uccello");
    r2 = r1;
  }
  public static void main (String[] args) {
    Q animale1 = new Q("Cane");
    Q animale2 = new Q("Gatto");
    m1(animale1,animale2);
    System.out.println(animale1.getName() + "," + animale2.getName());
}}
```

Qual è il risultato della compilazione ed esecuzione del programma ?

1. Stampa: Cane,Gatto
2. Stampa: Cane,Uccello
3. Stampa: Uccello,Gatto
4. Stampa: Uccello,Uccello
5. Errore di esecuzione
6. Errore di compilazione
7. Nessuna delle precedenti

27) Data la seguente classe:

```
class Q {
  private String name;
  public Q(String name) {this.name = name;}
  public void setName(String name) {this.name = name;}
  public String getName() {return name;}
  public static void m1(Q animale1, Q animale2) {
    animale1 = new Q("Pesce");
    animale2 = null;
  }
  public static void main (String[] args) {
    Q animale1 = new Q("Cane");
    Q animale2 = new Q("Gatto");
    m1(animale1,animale2);
    System.out.println(animale1.getName() + "," + animale2.getName());
}}
```

Qual è il risultato della compilazione ed esecuzione del programma ?

1. Stampa: Cane,Gatto
2. Stampa: Cane,Pesce
3. Stampa: Pesce,Gatto
4. Stampa: Pesce,Pesce
5. Errore di compilazione
6. Errore di esecuzione
7. Nessuna delle precedenti

28) Data la seguente classe:

```
class Q {
 static void m1(int[] i1, int[] i2) {
  int[] i3 = i1; i1 = i2; i2 = i3;
 }
 public static void main (String[] args) {
  int[] i1 = {1}, i2 = {3}; m1(i1, i2);
  System.out.print(i1[0] + "," + i2[0]);
}}
```

Qual è il risultato della compilazione ed esecuzione del programma ?

1. Stampa: 1,1
2. Stampa: 1,3
3. Stampa: 3,1
4. Stampa: 3,3
5. Errore di esecuzione
6. Errore di compilazione
7. Nessuna delle precedenti

29) Date le seguenti classi:

```
class A {
 void m1() {System.out.print("A.m1");}
}
class B extends A {
 void m1() {System.out.print("B.m1");}
 static void m1(String s) {System.out.print(s+",");}
}
class C {
 public static void main (String[] args) {
  B.m1("main");
  (new B()).m1();
}}
```

Qual è il risultato della compilazione ed esecuzione del programma ?

1. Stampa: main,B.m1
2. Errore di compilazione

3. Errore a Run-time
4. Nessuna delle precedenti

30) Quali delle seguenti affermazioni è vera ?

1. La relazione tra una classe e la superclasse è una relazione di composizione/aggregazione
2. La relazione tra una classe e la superclasse è una relazione di ereditarietà
3. La relazione tra una classe e un oggetto referenziato da un attributo della classe è una relazione di composizione/aggregazione
4. La relazione tra una classe e un oggetto referenziato da un attributo della classe è una relazione di ereditarietà

31) Date le seguenti classi:

```
class Zampa{}
abstract class Animale {
  public abstract void mangia() ;
  public abstract void dorme() ;
}
class Cane extends Animale {
  Zampa sinistraAnteriore;
  Zampa destraAnteriore;
  Zampa sinistraPosteriore;
  Zampa destraPosterire;
   Cane()
   {
   sinistraAnteriore=new Zampa();
   destraAnteriore=new Zampa();
   sinistraPosteriore=new Zampa();
   destraPosteriore=new Zampa();
   }
  public void mangia() {}
  public void dorme() {}
}
class Gatto extends Cane {
  public void disobbediente () {}
  public void siArrampicaSugliAlberi() {}
}
```

Quali delle seguenti affermazioni non è vera ?

1. Un Gatto eredita 4 istanze di Zampa dalla superclasse Cane
2. Un Gatto può mangiare e dormire
3. Un Gatto si arrampica sugli alberi
4. La relazione tra Cane e Animale è un esempio di uso approrpiato dell'ereditarietà
5. La relazione tra Gatto e Cane è un esempio di uso inappropriato dell'ereditarietà
6. Nessuna delle precedenti.

Soluzioni: Costrutti OO

Quesito	Soluzione	Commenti
1	2	Per il binding dinamico viene eseguita l'implementazione del metodo m1 della classe A
2	2	L'ordine di esecuzione è il seguente: • Invocazione del costruttore di A con un solo parametro (5) • Invocazione (tramite this) del costruttore di A con 2 parametri (5,10) • Invocazione del costruttore di B con il parametro di tipo float (5/4). Il cast a (int) rende 5/4 uguale a 1. Quindi z=3 • Esecuzione del metodo print 10+5+3
3	2	All'interno del metodo m1, per le regole sulla visibilità delle variabili, la prima istanza delle variabile x è l'attributo, la seconda è il valore della variabile dichiarata nel blocco interno al metodo m1 e definito dalle parentesi graffe.
4	3	L'errore di compilazione è dovuto al fatto che il tipo di dato B non definisce l'operazione m2. La variabile b punta però ad un oggetto il cui tipo di dato associato definisce l'operazione m2 (oggetto della classe A). Se si vuole invocare il metodo m2 si deve effettuare un cast e quindi sostituire all'istruzione b.m2();l'istruzione ((A) b).m2(); Quest'ultima istruzione non genera errori di compilazione.
5	4	La classe A deve implementare sia m1 che m2, altrimenti deve essere dichiarata astratta.
6	3	L'interfaccia Int2 ha i metodi m1 (ereditato) e m2. B dichiara di implementare Int2 quindi deve implementare sia m1 che m2 altrimenti si ha un errore di compilazione.
7	1	L'attributo x della classe A è di tipo private quindi non è visibile dalla classe B
8	1	Un attributo con modificatore di visibilità protected è visibile solo dalle sottoclassi.
9	4	Il metodo m1 della classe A è ricorsivo e chiama se stesso decrementando l'attributo x ad ogni chiamata fino a che x non diventa uguale a 0.
10	4	La sequenza delle operazioni è questa: • Viene creato un oggetto di tipo B (attributo y = 10) • L'oggetto di tipo B viene passato alla classe A • b.setValue(15) imposta y=15 • La prima stampa di a.getValue() restituisce 15 • a.m1() crea un altro oggetto di tipo B che assegna all'attributo b. Invocando di nuovo il costruttore di B ho nuovamente y=10 • La seconda stampa di a.getValue() restituisce 10
11	4	"void Q()" non è un costruttore perchè specifica il tipo ritornato (void) e nei costruttori questo non deve essere specificato. Dato che la classe Q ha già un costruttore "Q(int i)" non viene creato il costruttore di default. Quindi la classe non ha un costruttore a zero argomenti e quando viene invocato costruttore di questo tipo (riga 19) viene emesso un errore a tempo di compilazione.
12	1	La variabile b non è stata dichiarata

13	1	La variabile msg è visibile dalla classe Ered (friendly). Posso scrivere msg.text perché le classi sono tutte nello stesso package
14	1	La 2 non è corretta perché mancano le parentesi, la 3 non è corretta perché Fred eredita da due classi e in Java questo non è possibile,
15	3	• A riga 17 l'istruzione v.guida (); stampa: "Veicolo:guida" • A riga 18 l'istruzione c.guida(); stampa: "Automobile: guida" (nella classe Automobile viene fatto l'override del metodo guida) • A riga 19 l'istruzione v=c (possibile per il polimorfismo) fa si che il riferimento v punti allo stesso oggetto puntato dal riferimento c. Quindi v punta ad un oggetto di tipo Automobile. • A riga 20 l'istruzione v.guida(); provoca l'invocazione del metodo guida di Automobile() (binding dinamico) e quindi la stampa di "Automobile: guida"
16	2	Se non si inserisce come prima istruzione viene restituito un errore a tempo di compilazione.
17	3	A partire dalla linea 6 (*e=null*) il riferimento *e* non punta più all'oggetto considerato. Inoltre non esistendo, oltre ad *e*, nessun altro riferimento a tale oggetto questo può essere distrutto dal garbage collector che libera la memoria occupata.
18	1,2,4	• La classe A eredita implicitamente dalla classe Object. Questo vale in Java per tutte le classi che non ereditano esplicitamente da una classe. Infatti tutte le classi (anche le proprie) ereditano direttamente o indirettamente dalla classe Object. • Il compilatore inserisce un costruttore di default se la classe non ne definisce uno. • Il costruttore di default invoca il costruttore a zero argomenti della superclasse.
19	4	L'errore di compilazione a riga 2 deriva dal fatto che il compilatore inserisce un costruttore di default nella classe B. Tale costruttore invoca il costruttore a zero argomenti della superclasse. Dato che nella superclasse un costruttore di questo tipo non esiste, viene restituito un errore di compilazione a riga 2.
20	1,2,3	
21	3	Il modificatore di visibilità "private" può essere specificato per la dichiarazione degli attributi di una classe; non ha senso per le variabili locali di un metodo.
22	4	• r1.a vale 1 perché l'attributo a dell'oggetto r1 è stato incrementato dall'istruzione r1.a++ • r1.b vale 1 perché l'attributo statico b della classe Rosso è stato incrementato dall'istruzione r1.b++ • r2.a vale 0 perché l'attributo a dell'oggetto r1 non è stato ma inizializzato • r2.b vale 1 ed è lo stesso valore di r1.b dato che b è statico ed è un attributo della classe Rosso e non dei suoi oggetti. Quindi tutti gli oggetti della classe Rosso "vedono" lo stesso valore.
23	1,2	Una classe final non può essere ereditata e quindi i suoi metodi non possono essere sovrascritti ed in questo senso sono implicitamente final.
24	2	