

# LP1 – Prova Intermedia 1

*prof. Eliana Minicozzi*

*prof. Marcello Sette*

22 aprile 2006

## Esercizio 1 – max 10/30

Sia dato il programma Pascal:

```
program esercizio1(input, output);
var
  a, b, c: integer;

procedure p1(var a, b: integer);
begin
  c:= a;
  if b<a then
    a:=1 else
    b:=7
end;

procedure p2(b: integer;
            var c: integer);
procedure p3(c: integer;
            var a: integer);
begin
  b:= a+3; c:= 3;
  p1(c, b);
  writeln(a, b, c)
end;

begin
  a:= c+2; b:= 2;
  p3(c, b);
  writeln(a, b, c)
end;

begin
  a:= 0; b:= 4; c:= 8;
  p2(b, a);
  writeln(a, b, c)
end.
```

Sapendo che il linguaggio usa ambito statico di validità dei nomi e che, nelle procedure, i parametri **var** sono parametri INOUT realizzati per riferimento, quelli non-**var** sono parametri IN realizzati per copia,

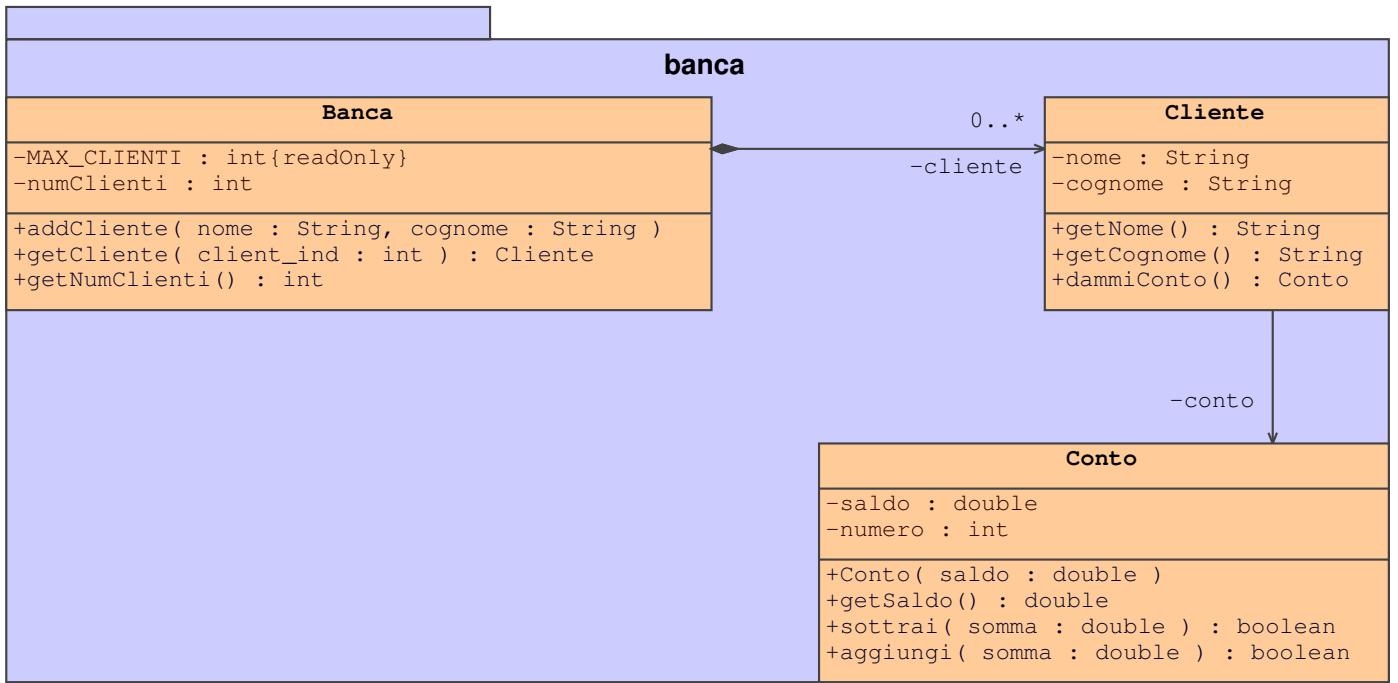
1. rappresentare lo stack di esecuzione quando esso ha altezza massima;
2. specificare le variabili locali e non locali di ogni procedura;
3. discutere il comportamento del programma ed il suo output;
4. (facoltativo) ripetere lo stesso esercizio supponendo, invece, che i riferimenti INOUT siano realizzati per copia.

## Esercizio 2 – max 5/30

Rappresentare graficamente tutti i Data Object e i legami coinvolti nell'esecuzione di questo breve programma Pascal:

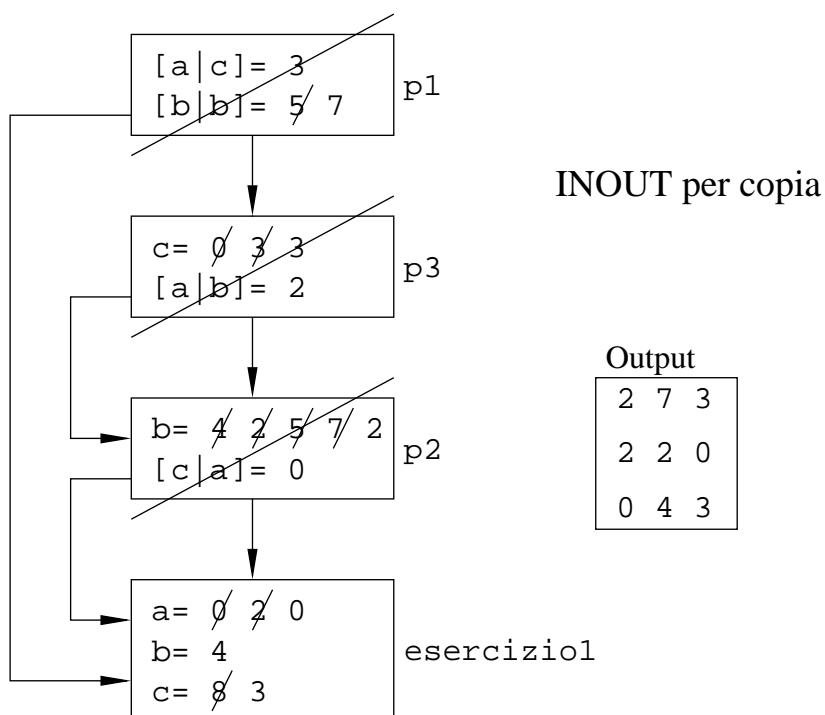
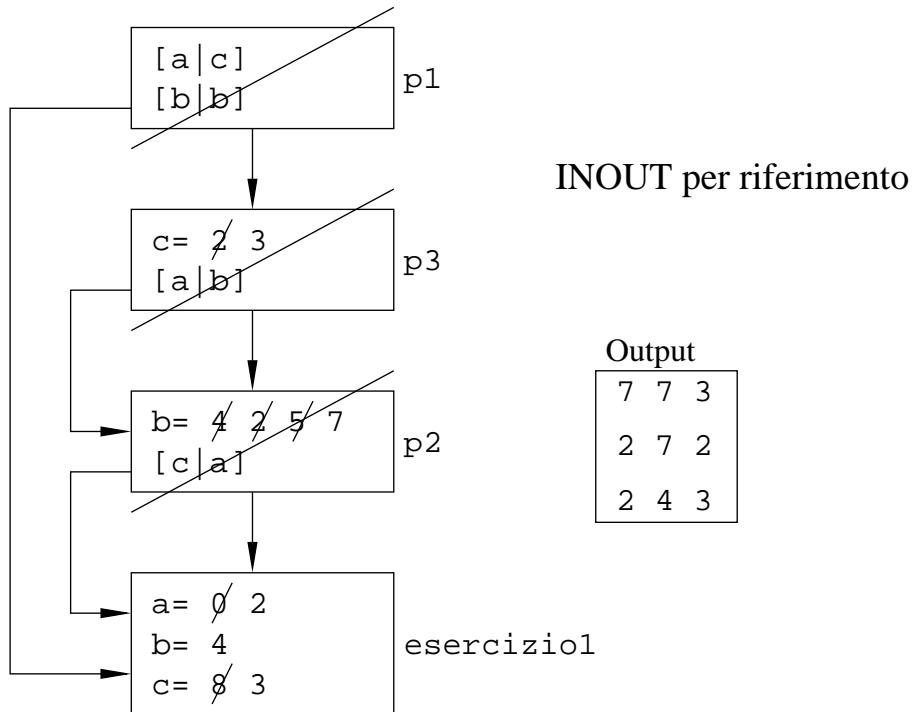
```
program esercizio2 (input, output);
var
  x: integer;
  p: ^integer;
begin
  x:= 7;
  new(p);
  p^:= x;
  writeln(p^)
end.
```

## Esercizio 3 – max 15/30



1. Descrivere dettagliatamente il precedente diagramma UML, specificando il tipo di diagramma ed il significato dei simboli presenti, con particolare riguardo alle componenti delle classi ed alla loro visibilità.
  2. Modificare il diagramma precedente, sapendo che:
    - (a) Nella classe **Conto**, l'attributo `saldo` è **protected**.
    - (b) Ogni cliente può avere fino a 4 conti in banca.
    - (c) Vi sono due tipologie di conto: una, il libretto di risparmio, in cui viene concesso un tasso di interesse creditore; l'altra, il conto corrente, in cui viene concesso uno scoperfo, cioè la possibilità di avere anche saldo negativo fino ad un certo limite inferiore.
    - (d) Nel conto corrente viene anche concessa la possibilità che una richiesta di sottrazione fondi, qualora superi la somma tra saldo e scoperfo, sia eventualmente soddisfacibile con i fondi di un altro conto (conto *fideiussore*), anche intestato ad altra persona.
  3. Sapendo che:
    - (a) Eliana Minicozzi, Marcello Sette, Renato Musto sono clienti della BNL.
    - (b) Eliana possiede un libretto di risparmio con un saldo di 2000,00 euro e tasso di 0,05.
    - (c) Marcello possiede un conto corrente con un saldo di 1000,00 euro e 200,00 euro di scoperfo.
    - (d) Renato possiede un conto corrente con saldo di 300,00 euro, scoperfo di 200,00 euro, fideiussione al conto di Eliana.
- rappresentare in diagrammi di sequenza i seguenti scenari di casi d'uso:
- (a) Un utente esterno chiede a Marcello di poter agire sul suo primo conto. Avuto il conto, l'utente prima deposita su di esso 200,00 euro, poi preleva 1500,00 euro. La seconda operazione fallisce.
  - (b) Un utente esterno chiede a Renato di poter agire sul suo primo conto. Avuto il conto, l'utente prima deposita su di esso 200,00 euro, poi preleva 1500,00 euro. La seconda operazione, non soddisfacibile direttamente, richiede il prelievo della differenza dal conto fideiussore.

## Soluzione esercizio 1

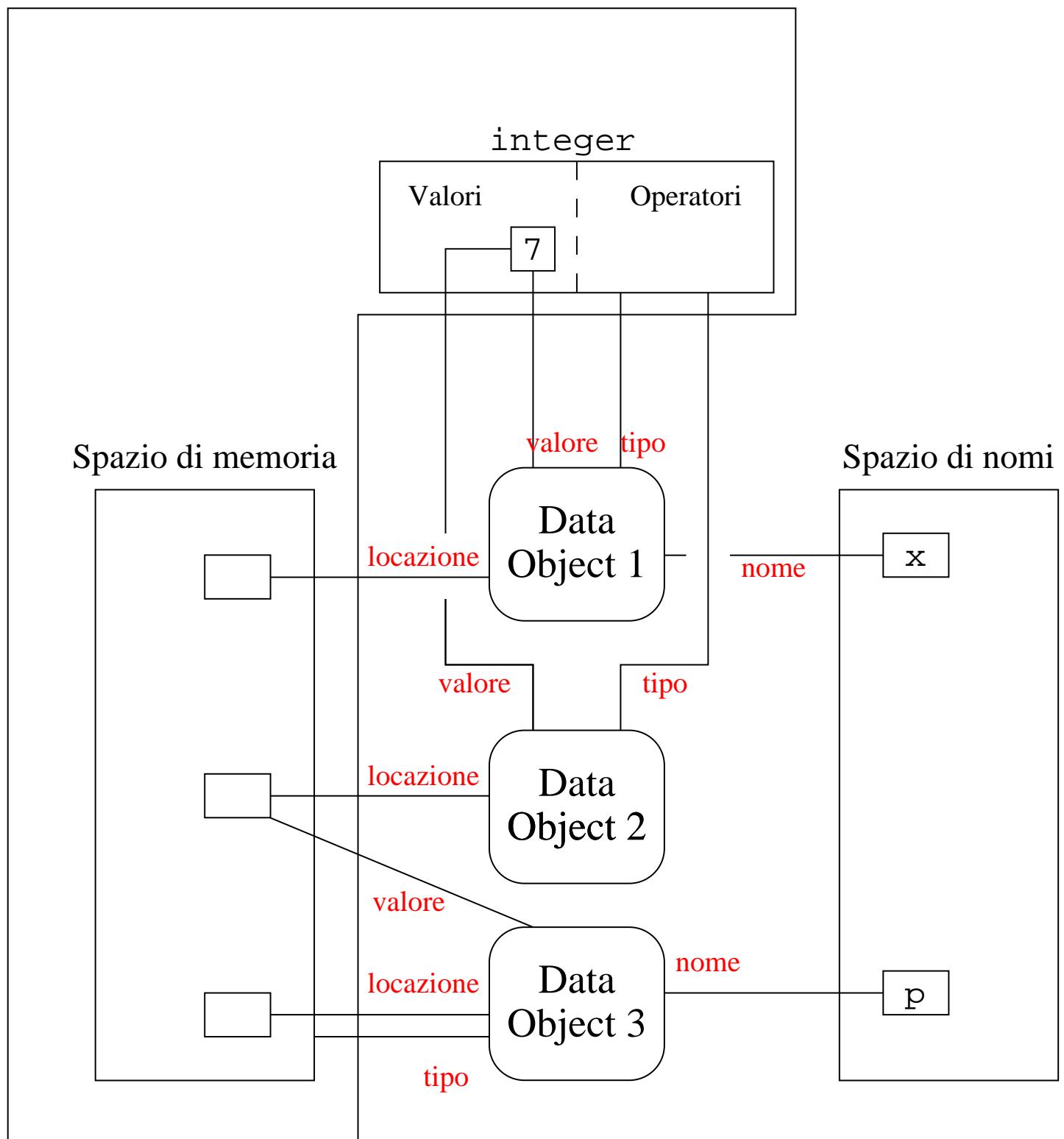


Legenda:

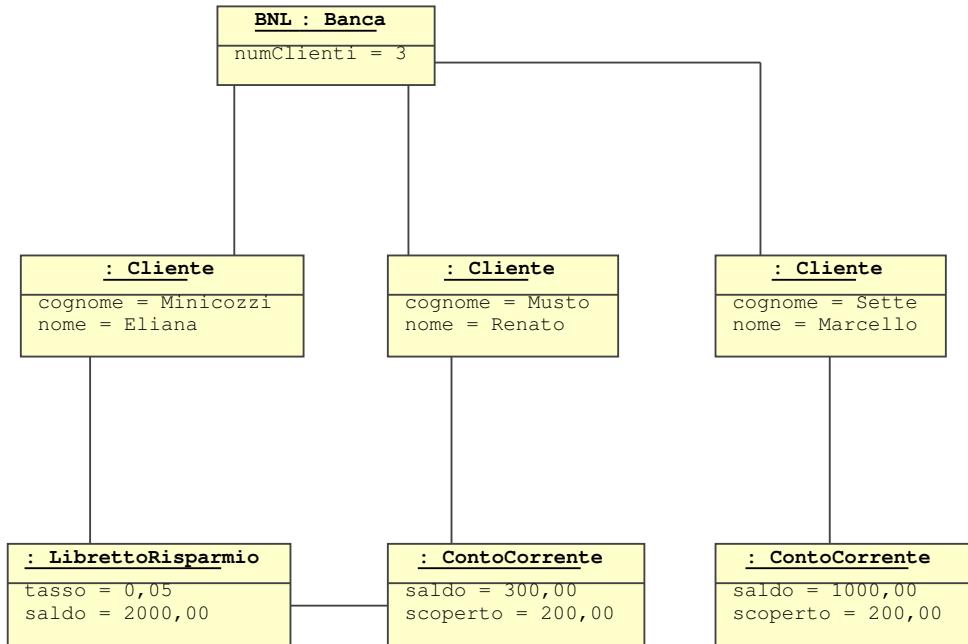
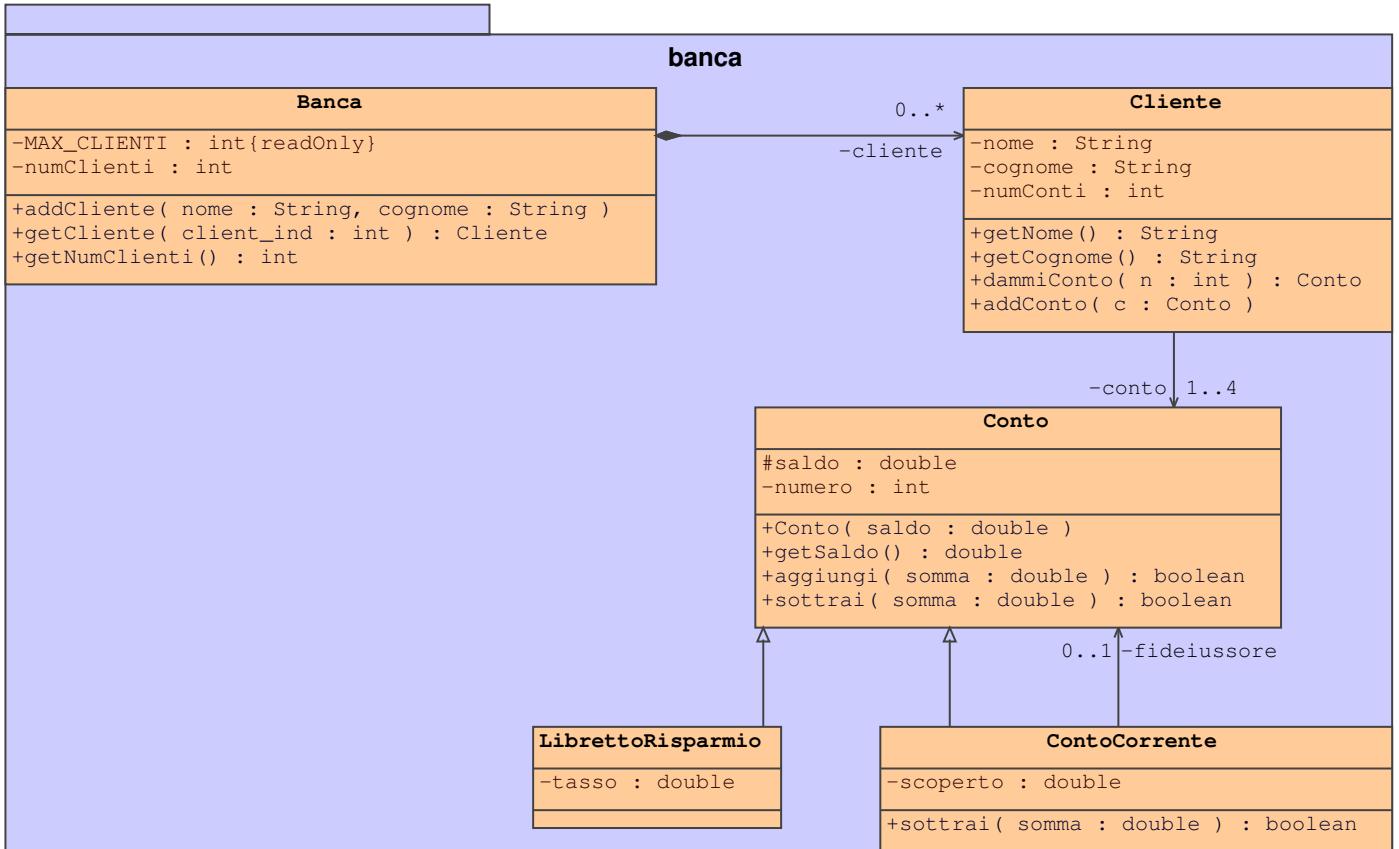
- $[a|b]$ : parametro formale  $a$ , parametro attuale  $b$ , passaggio INOUT per riferimento.
- $[a|b]=4$ : parametro formale  $a$ , parametro attuale  $b$  con valore 4, passaggio INOUT per copia.
- $a=4$ : parametro formale  $a$ , valore del parametro attuale 4, passaggio IN per copia.

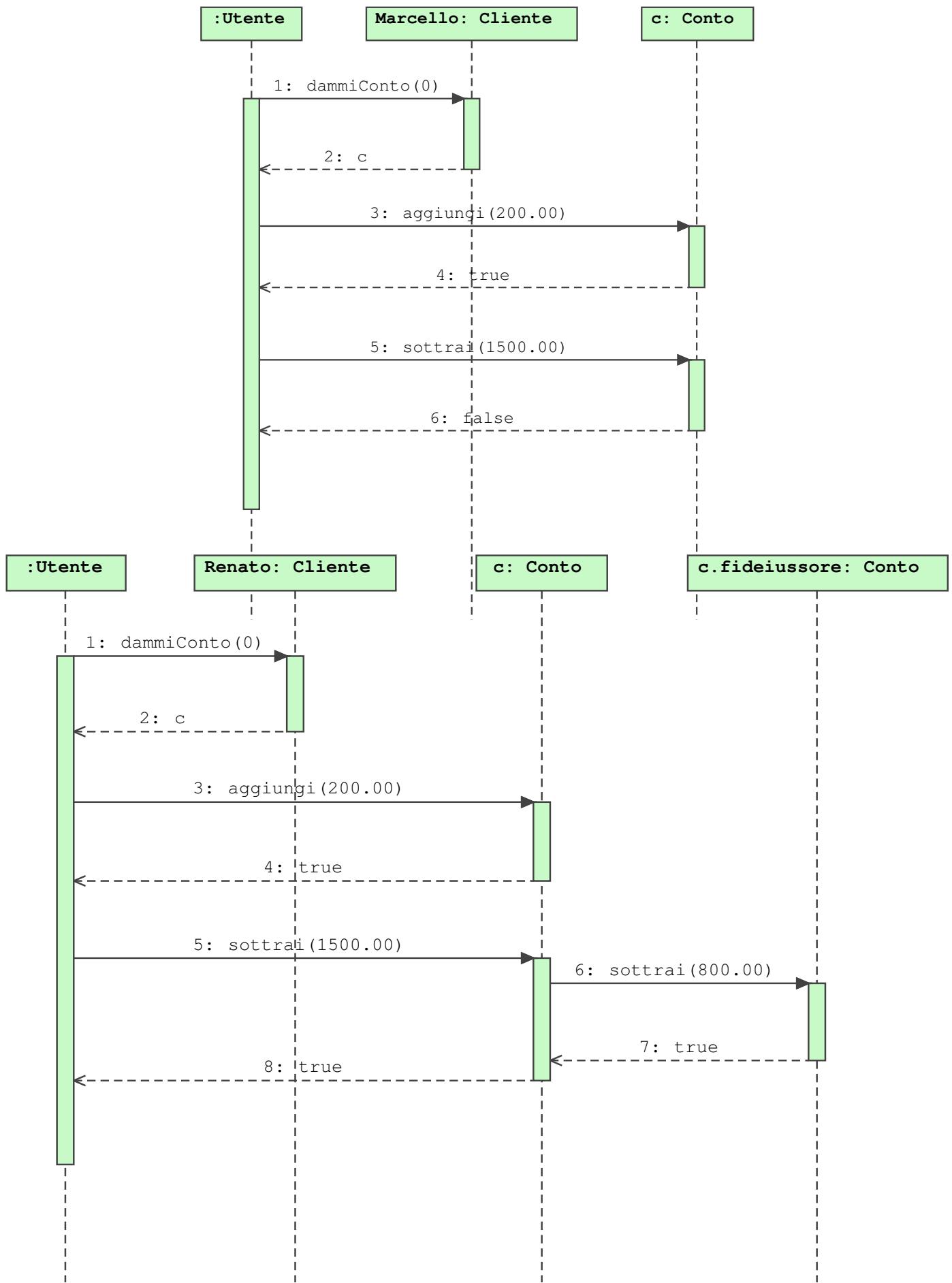
## Soluzione esercizio 2

### Spazio di tipi



# Soluzione esercizio 3





Segnare nell'ultimo foglio le risposte alle seguenti domande a scelta multipla, tenendo presente che alcune domande potrebbero richiedere una risposta multipla.

1. È possibile creare array di lunghezza zero? (Una risposta)

- A. Sì, di qualunque tipo.
- B. Sì, ma solo di tipi primitivi.
- C. Sì, ma solo di riferimenti ad oggetti.
- D. No, tranne per l'array ricevuto dal metodo main contenente la lista degli argomenti della riga di comando.
- E. No, in nessun caso.

2. Quanto è restrittiva l'accessibilità di default? (Una risposta)

- A. Meno restrittiva di `public`.
- B. Più restrittiva di `public`, ma meno restrittiva di `protected`.
- C. Più restrittiva di `protected`, ma meno restrittiva di `private`.
- D. Più restrittiva di `private`.
- E. Meno restrittiva di `protected` dall'interno di un pacchetto, ma più restrittiva di `protected` dall'esterno di un pacchetto.

3. Cosa c'è di sbagliato nel codice seguente? (Una risposta)

```
abstract class A implements I1, I2 {
    public void f() {}
    public void g() {}
}

interface I1 {
    int VAL_A = 1;
    int VAL_B = 2;

    void f();
    void g();
}

interface I2 {
    int VAL_B = 3;
```

```
int VAL_C = 4;
```

```
void g();
void h();
}
```

- A. La classe A non può implementare due interfacce.
- B. La classe A non ha fornito l'implementazione del metodo `h()` dell'interfaccia I2.
- C. C'è un conflitto delle dichiarazioni dei due metodi `g()` nelle interfacce I1 e I2.
- D. C'è un conflitto delle dichiarazioni degli attributi `VAL_B` nelle interfacce I1 e I2.
- E. Non c'è nulla di sbagliato.

4. Quale delle seguenti affermazioni è vera? (Una risposta)

- A. Un programma non esaurirà mai la memoria.
- B. Gli oggetti che non saranno mai più usati sono eleggibili per la garbage collection.
- C. Gli oggetti che sono riferiti da altri oggetti non sono mai eleggibili per la garbage collection.
- D. Gli oggetti che possono essere raggiunti da altri oggetti attivi non sono eleggibili per la garbage collection.
- E. Gli oggetti sono immediatamente deallocati non appena il sistema riconosce che sono eleggibili per la garbage collection.

5. Qual è il risultato del seguente programma? (Una risposta)

```
public class X {
    public static void main(String args[]) {
        try {
            cattivo();
            System.out.print("A");
        }
        catch(RuntimeException e) {
            System.out.print("B");
        }
        catch(Exception ex) {
            System.out.print("C");
        }
        finally {
            System.out.print("D");
        }
        System.out.print("E");
    }

    public static void cattivo() {
        throw new RuntimeException();
    }
}
```

- A. BD

- B. BCD
- C. BDE
- D. BCDE
- E. Errore di compilazione.

6. Quali delle seguenti affermazioni sono vere? (Tre risposte)

- A. Il costruttore di default inizializza le variabili dei metodi.
- B. Il costruttore di default ha lo stesso livello di accesso della sua classe.
- C. Il costruttore di default invoca `super()`.
- D. Se la classe non contiene un costruttore senza argomenti, il compilatore crea sempre un costruttore di default.
- E. Il compilatore crea un costruttore di default solo se non vi sono altri costruttori nella classe.

7. Quale delle seguenti è la definizione legale di una classe che non può essere istanziata? (Una risposta)

- A. `class Ghost { abstract void h(); }`
- B. `abstract class Ghost { void h(); }`
- C. `abstract class Ghost { void h() {} }`
- D. `abstract Ghost { abstract void h(); }`
- E. `static class Ghost { abstract h(); }`

8. Qual è il risultato dell'esecuzione del seguente programma? (Una risposta)

```
public class MiaClasse {
    public static void main(String args[]) {
        B b = new B("Test");
    }
}
```

```
class A {
    A() { this("1", "2"); }

    A(String s, String t) { this(s + t); }

    A(String s) { System.out.println(s); }
}
```

```
class B extends A {
    B(String s) { System.out.println(s); }
```

```
    B(String s, String t) {
        this(t + s + "3");
    }

    B() { super("4"); }
}
```

- A. Solo Test
- B. Test seguito da Test.
- C. 123 seguito da Test.
- D. 12 seguito da Test.
- E. 4 seguito da Test.

9. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
public class P {
    public static void main(String[] args) {
        A ref1 = new C();
        B ref2 = (B) ref1;
        System.out.println(ref2.g());
    }
}
```

```
class A {
    private int f() { return 0; }
    public int g() { return 3; }
}
```

```
class B extends A {
    private int f() { return 1; }
    public int g() { return f(); }
}
```

```
class C extends B {
    public int f() { return 2; }
}
```

- A. 0
- B. 1
- C. 2
- D. 3
- E. Errore di compilazione.

10. Dato:

1. `public class Esterna {`
2. `private double d1 = 1.0;`
3. `// inserire qui il codice`
4. `}`

Quali brani di codice sono validi se inseriti singolarmente alla linea 3? (Due risposte)

```

A. static class Interna {
    public double m() { return d1; }
}

B. static class Interna {
    static double m() { return d1; }
}

C. private class Interna {
    public double m() { return d1; }
}

D. protected class Interna {
    static double m() { return d1; }
}

E. public abstract class Interna {
    public abstract double m();
}

```

---

11. Qual è il risultato del seguente programma? (Una risposta)

```

public class X {
    public static void main(String args[]) {
        try {
            cattivo();
            System.out.print("A");
        }
        catch(Exception ex) {
            System.out.print("B");
        }
        finally {
            System.out.print("C");
        }
        System.out.print("D");
    }

    public static void cattivo() {}
}

```

- A. AC
  - B. BD
  - C. ACD
  - D. AD
  - E. Errore di compilazione.
- 

12. Date le seguenti coppie di dichiarazioni di metodi, quali delle successive affermazioni sono vere? (Due risposte)

```

void fly(int d) {}
int  fly(int t, int s) { return t*s; }

void fall(int t) {}
int  fall(int d) { return d; }

void glide(int t) {}
void Glide(int t) {}

```

- A. La prima coppia di metodi è un sovraccaricamento legale.
  - B. La seconda coppia di metodi è un sovraccaricamento legale.
  - C. La terza coppia di metodi è un sovraccaricamento legale.
  - D. La seconda coppia di metodi non viene compilata correttamente.
  - E. La terza coppia di metodi non viene compilata correttamente.
- 

13. Dato il codice seguente, quale delle linee che sono state commentate può essere reinserita senza causare errori? (Una risposta)

```

abstract class Mia {
    abstract void f();
    final   void g() {}
// final   void h() {}           // (1)

    protected static int i;
    private      int j;
}

final class Tua extends Mia {
    int m;
// Tua(int n) { m=n; }         // (2)

    public static void main(String args[]) {
        Tua t = new Tua();
    }

    void f() {}
    void h() {}
// void k() { i++; }           // (3)
// void l() { j++; }           // (4)
}

```

- A. La linea etichettata (1).
  - B. La linea etichettata (2).
  - C. La linea etichettata (3).
  - D. La linea etichettata (4).
  - E. Nessuna linea.
- 

14. Qual è la prima linea che causa un errore di compilazione nel seguente programma? (Una risposta)

```

class MiaClasse {
    public static void main (String[] args) {
        MiaClasse a;
        MiaSottoclaasse b;

        a = new MiaClasse();
        b = new MiaSottoclaasse();           // (1)
    }
}

```

```

    a = b;                      // (2)           System.out.print("A");
    b = a;                      // (3)
    }
}

class MiaSottoclasse extends MiaClasse {}
```

- A. La linea etichettata (1).
  - B. La linea etichettata (2).
  - C. La linea etichettata (3).
  - D. La linea etichettata (4).
  - E. La linea etichettata (5).
- 

15. Qual è il risultato del seguente programma? (Una risposta)

```

public class X {
    public static void main(String args[]) {
        try {
            cattivo();
            System.out.print("A");
        }
        catch(RuntimeException e) {
            System.out.print("B");
        }
        catch(Exception ex) {
            System.out.print("C");
        }
        finally {
            System.out.print("D");
        }
        System.out.print("E");
    }

    public static void cattivo() {
        throw new Exception();
    }
}
```

- A. CD
  - B. CDE
  - C. ACDE
  - D. ACD
  - E. Errore di compilazione.
- 

16. Qual è il risultato del seguente programma? (Una risposta)

```

public class X {
    public static void main(String args[]) {
        try {
            cattivo();
```

```

                System.out.print("A");
            }
            catch(Exception ex) {
                System.out.print("B");
            }
            finally {
                System.out.print("C");
            }
            System.out.print("D");
        }
```

```

    public static void cattivo() {
        throw new RuntimeException();
    }
}
```

- A. AB
  - B. BC
  - C. ABC
  - D. BCD
  - E. Errore di compilazione.
-

**LP1 – Prova d'esame****Foglio delle risposte**

proff. Eliana Minicozzi e Marcello Sette

18 giugno 2007

Studente e matricola:

Ora di inizio:

Ora di consegna:

---

|    |   |   |   |   |   |
|----|---|---|---|---|---|
| 1  | A | B | C | D | E |
| 2  | A | B | C | D | E |
| 3  | A | B | C | D | E |
| 4  | A | B | C | D | E |
| 5  | A | B | C | D | E |
| 6  | A | B | C | D | E |
| 7  | A | B | C | D | E |
| 8  | A | B | C | D | E |
| 9  | A | B | C | D | E |
| 10 | A | B | C | D | E |
| 11 | A | B | C | D | E |
| 12 | A | B | C | D | E |
| 13 | A | B | C | D | E |
| 14 | A | B | C | D | E |
| 15 | A | B | C | D | E |
| 16 | A | B | C | D | E |

**Valutazione della prova 1:**

Segnare nell'ultimo foglio le risposte alle seguenti domande a scelta multipla, tenendo presente che alcune domande potrebbero richiedere una risposta multipla.

1. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
class EsSuper {
    String prima, seconda;
    public EsSuper(String p, String s) {
        prima=p;
        seconda=s;
    }
    public String toString() {
        return prima + " " ;
    }
}

public class Esempio extends EsSuper {
    public Esempio(String p, String s) {
        super(p, s);
    }
    public String toString() {
        return prima + ":" + seconda + " " ;
    }
    public static void main(String args[]) {
        EsSuper a = new EsSuper("A", "a");
        EsSuper b = new Esempio("B", "b");

        System.out.print(a.toString());
        System.out.print(b.toString());
    }
}
```

- A. Errore in compilazione.
- B. Errore in esecuzione.
- C. A B
- D. A B:b
- E. A:a B:b

2. Data la classe seguente:

```
class Esempio {
    private int x;
    public static void main(String[] args) {
        // in questo contesto
    }
}
```

Quale delle seguenti affermazioni è vera? (Una risposta)

- A. Se si cambia il modificatore di x da `private` a `public`, allora si può usare `this.x` nel contesto del metodo `main`.

- B. Se si elimina il modificatore `private` di x, allora si può usare `this.x` nel contesto del metodo `main`.
- C. Se si cambia il modificatore `private` di x, da `private` a `static`, allora si può usare `this.x` nel contesto del metodo `main`.
- D. Se si aggiunge il modificatore `public` alla classe `Esempio`, allora si può usare `this.x` nel contesto del metodo `main`.
- E. Non si può mai usare `this.x` nel contesto del metodo `main`.

3. Quali dei seguenti enunciati definiscono correttamente una variabile adatta a riferire un array di 50 oggetti `String`? (Tre risposte)

- A. `String a[50];`
- B. `Object a;`
- C. `String[] a;`
- D. `Object a[];`
- E. `char a[][];`

4. Dato il seguente metodo:

```
public void metodo() {
    try {
        delicato();
        System.out.println("Punto 1");
    }
    catch(ArrayIndexOutOfBoundsException e) {
        System.out.println("Punto 2");
    }
    finally {
        System.out.println("Punto 3");
    }
    System.out.println("Punto 4");
}
```

Indicare tutti gli output che saranno prodotti se il metodo `delicato()` termina normalmente senza lanciare alcuna eccezione.

- A. Punto 1
- B. Punto 2
- C. Punto 3
- D. Punto 4
- E. Nessun output.

5. Quale delle seguenti affermazioni è vera? (Una risposta)

- A. La garbage collection richiede ulteriore codice nel caso in cui possono essere lanciate eccezioni.
- B. Il programmatore può indicare che un riferimento, realizzato mediante una variabile locale, non sarà più utilizzato.
- C. Il programmatore ha la possibilità di liberare, immediatamente ed esplicitamente, la memoria usata da un oggetto.
- D. La garbage collection viene eseguita in tempi predibili.
- E. La garbage collection non libera mai la memoria occupata da oggetti che sono ancora accessibili dallo stack di attivazione.

---

6. Date le seguenti dichiarazioni:

```
class C {  
    private long val;  
    public C (long v) {  
        val = v;  
    }  
}  
  
C x = new C(10L);  
C y = new C(10L);  
C z = y;  
long a = 10L;  
int b = 10;
```

Quali espressioni logiche sono valide e hanno valore **true**? (Tre risposte)

- A. (a == b)
  - B. (a == x)
  - C. (y == z)
  - D. (x == y)
  - E. (a == 10.0)
- 

7. Dato un file contenente:

```
// PUNTO INDICATO  
public class Buono {}
```

Quali righe potrebbero essere alternativamente e legalmente aggiunte al PUNTO INDICATO? (Due risposte)

- A. protected class Migliore {}
  - B. class Migliore {}
  - C. public class Migliore {}
  - D. public static final int OTTIMO=7;
  - E. import java.awt.\*;
- 

8. Assumendo in ciascun caso che il codice sia l'intero contenuto di un file di nome **Marc.java** in un file system case-sensitive, quali delle seguenti sono definizioni corrette di una classe? (Tre risposte)

- A. public class Marc {  
 public int x=0;  
 public Marc(int xval) {  
 x=xval;  
 }  
}
  - B. public class MARC {  
 public int x=0;  
 public MARC(int xval) {  
 x=xval;  
 }  
}
  - C. public class Marc extends Ello {  
 public int x=0;  
 public Marc(int xval) {  
 x=xval;  
 }  
}
- 

```
D. static class Marc {  
    static private int x=0;  
    static private Marc(int xval) {  
        x=xval;  
    }  
}  
  
E. import java.io.*;  
public class Marc {  
    private int x=0;  
    public Marc(int xval) {  
        x=xval;  
    }  
}
```

---

9. Un vincolo del nostro progetto richiede che una particolare variabile di istanza non sia accessibile in alcun modo al codice all'esterno del pacchetto al quale appartiene la classe. Come si può realizzare, nel modo meno restrittivo possibile, tale vincolo? (Una risposta)

- A. La variabile deve essere marcata **public**.
  - B. La variabile deve essere marcata **private**.
  - C. La variabile deve essere marcata **protected**.
  - D. La variabile non deve avere alcun modificatore di accesso.
  - E. La variabile deve essere marcata **private** e deve essere fornito un metodo che restituisca il valore della variabile.
- 

10. Dato il seguente corpo di un **metodo**:

```
{  
    if (test()) {  
        pericoloso();  
    } else {  
        innocuo();  
    }  
}
```

Il metodo **pericoloso()** potrebbe lanciare una **ProtocolException** (che non è sottoclassificata di **RuntimeException**). Quali potrebbero essere due dichiarazioni corrette del **metodo**? (Due risposte)

- A. public metodo()
  - B. public void metodo()
  - C. public void metodo() throw IOException
  - D. public void metodo() throws ProtocolException
  - E. public void metodo() throws Exception
- 

11. Qual è il risultato del seguente programma? (Una risposta)

```
public class Test {  
    public static void main(String args[]) {  
        int[] x = new int[10];  
        System.out.println(x[5]);  
    }  
}
```

- A. Viene lanciata una **ArrayIndexOutOfBoundsException**.
- B. Viene lanciata una **NullPointerException**.

- C. Errore di compilazione per una errata dichiarazione di un array.
- D. Errore di compilazione diverso dal precedente.
- E. 0

12. Le due classi seguenti sono definite in file separati:

```
public class Base
    public int b;
    public void m_a(float f) {
        int a;
    }
}

public class Sub extends Base {
    private float c;
    public void m_b() {
        Base ba = new Base();
        int i;
        float g;
        // PUNTO INDICATO
    }
}
```

Quali dei seguenti enunciati sono validi al PUNTO INDICATO? (Tre risposte)

- A. `i=ba.a;`
- B. `i=ba.b;`
- C. `g=c;`
- D. `g=ba.f;`
- E. `ba.m_a(6.2F);`

13. Quale linea del codice seguente causa un errore di compilazione? (Una risposta)

```
1. class Sub extends Base {}
2.
3. class Base {
4.     String s;
5.     public Base() {}
6.     public Base(String st) {
7.         s=st;
8.     }
9. }

10.
11. public class Esempio {
12.     public void metodo() {
13.         Sub s = new Sub("A");
14.         Base b = new Base("B");
15.     }
16. }
```

- A. La linea 14.
- B. La linea 3.
- C. La linea 6.
- D. La linea 13.
- E. Nessuna linea.

14. Quale modificatore dovrebbe essere applicato alla dichiarazione di una variabile membro di una classe affinché il valore della variabile rimanga costante dopo la creazione di ogni oggetto? (Una risposta)

- A. `final`
- B. `static`
- C. `abstract`
- D. `private`
- E. `protected`

15. Sia dato il seguente metodo incompleto:

```
1. public void metodo() {
2.
3.     if (predicato()) {
4.
5.     }
6.
7. }
```

Si vuole che il metodo lanci una `IOException` (sottoclasse diretta di `Exception`) se e solo se il `predicato()` restituisce un valore `true`. Mediante quali due dei seguenti interventi si può ottenere il risultato voluto? (Due risposte)

- A. Aggiungendo alla linea 2: `IOException e;`
- B. Aggiungendo alla linea 4: `throw e;`
- C. Aggiungendo alla linea 4: `throw new IOException();`
- D. Aggiungendo alla linea 6: `throw new IOException();`
- E. Modificando la dichiarazione del metodo per indicare che potrebbe essere lanciato un oggetto di tipo `Exception`.

16. Dato il seguente metodo:

```
1. public void metodo(String s) {
2.     String a, b;
3.     a = new String("Ciao");
4.     b = new String("Addio");
5.     System.out.println(a+b);
6.     a = null;
7.     a = b;
8.     System.out.println(a+b);
9. }
```

Qual è il punto in cui l'oggetto costruito alla riga 3 diventa eleggibile per la garbage collection? (Una risposta)

- A. Subito dopo la riga 4.
- B. Subito dopo la riga 5.
- C. Subito dopo la riga 6.
- D. Subito dopo la riga 7.
- E. Mai in questo metodo.

**LP1 – Prova d'esame****Foglio delle risposte**

proff. Eliana Minicozzi e Marcello Sette

20 luglio 2007

Studente e matricola:

Gruppo [1:Minicozzi, 2:Sette]:

|    |   |   |   |   |   |
|----|---|---|---|---|---|
| 1  | A | B | C | D | E |
| 2  | A | B | C | D | E |
| 3  | A | B | C | D | E |
| 4  | A | B | C | D | E |
| 5  | A | B | C | D | E |
| 6  | A | B | C | D | E |
| 7  | A | B | C | D | E |
| 8  | A | B | C | D | E |
| 9  | A | B | C | D | E |
| 10 | A | B | C | D | E |
| 11 | A | B | C | D | E |
| 12 | A | B | C | D | E |
| 13 | A | B | C | D | E |
| 14 | A | B | C | D | E |
| 15 | A | B | C | D | E |
| 16 | A | B | C | D | E |

**Valutazione della prova 1:**

## LP1

## Prova d'esame

proff. E. Minicozzi e M. Sette

13 settembre 2007

Segnare nell'ultimo foglio le risposte alle seguenti domande a scelta multipla, tenendo presente che alcune domande potrebbero richiedere una risposta multipla.

1. Quali sono i due sovraccaricamenti corretti del costruttore della classe seguente? (Due risposte)

```
public class A {
    public A(int x, int y) {}
```

- A. A() {}
- B. protected int A() {}
- C. private A(int x, byte y) {}
- D. public Object A(int x, int y) {}
- E. public void A(byte x, byte y) {}

2. Qual è il risultato della compilazione del codice seguente? (Una risposta)

```
1. interface Foo {
2.     int k = 0;
3. }
4.
5. public class Test implements Foo {
6.     public static void main(String args[]) {
7.         int i;
8.         Test test = new Test();
9.         i = test.k;
10.        i = Test.k;
11.        i = Foo.k;
12.    }
13. }
```

- A. La compilazione ha successo.
- B. Errore di compilazione alla linea 2.
- C. Errore di compilazione alla linea 9.
- D. Errore di compilazione alla linea 10.
- E. Errore di compilazione alla linea 11.

3. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
class Super {
    public int i = 0;
    public Super (String text) {
        i = 1;
    }
}

public class Sub extends Super {
    public Sub (String text) {
        i = 2;
    }
    public static void main (String args[]) {
        Sub sub = new Sub("Ciao");
        System.out.println(sub.i);
    }
}
```

- A. Errore di compilazione.
- B. 0
- C. 1
- D. 2
- E. Errore di esecuzione.

4. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
public class Test {
    public static String output = "";
    public static void foo(int i) {
        try {
            if(i==1) {
                throw new Exception();
            }
            output += "1";
        }
        catch(Exception e) {
            output += "2";
            return;
        }
        finally {
            output += "3";
        }
        output += "4";
    }
}

public static void main (String args[]) {
    foo(0);
    foo(1);
    System.out.println(output);
}
```

- A. 1341234
- B. 134234
- C. 13423
- D. 134123

5. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
class G {
    String s1 = "G";
    void printS1() {
        System.out.print(s1);
    }
    G() {
        printS1();
    }
}

class H extends G {
    String s1 = "H";
    void printS1() {
        System.out.print(s1);
    }
    public static void main(String[] args) {
        H h = new H();
    }
}
```

- A. H
- B. G
- C. null
- D. Errore di compilazione o di esecuzione.
- E. Il programma viene correttamente compilato ed eseguito, ma non stampa nulla.

6. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
class Blue {
    Blue() {
        System.out.print("A");
    }
    static {
        System.out.print("B");
    }
    public static void main(String args[]) {
        System.out.print("C");
        Blue blue = new Blue();
    }
}
```

- A. CBA
- B. CAB
- C. BCA
- D. BAC
- E. Un risultato diverso dai precedenti.

7. Dato un file contenente:

```
package foo;

public class Outer {
    public static class Inner {
    }
}
```

Quale delle seguenti affermazioni è vera? (Una risposta)

- A. Una istanza della classe `Inner` può essere costruita con “`new Outer.Inner()`”.
- B. Una istanza della classe `Inner` non può essere costruita all'esterno del pacchetto `foo`.
- C. Una istanza della classe `Inner` può essere costruita solo all'interno della classe `Outer`.
- D. All'interno del pacchetto `bar`, una istanza della classe `Inner` può essere costruita con “`new Inner()`”.
- E. Nessuna delle precedenti è vera.

8. Dato il codice seguente:

```
1. class Super {
2.     public float getNum() {return 3.0f;}
3. }
4.
5. public class Sub extends Super {
6.
7. }
```

Quale metodo, posto alla linea 6, causerà un errore di compilazione? (Una risposta)

- A. `public float getNum() {return 4.0f;}`
- B. `public void getNum() {}`
- C. `public void getNum(double d) {}`
- D. `public double getNum(float d) {return d;}`
- E. Nessuno dei precedenti.

9. Sapendo che `java.io.IOException` è sottoclasse di `Exception` ma non di `RuntimeException`, qual è il risultato della compilazione ed esecuzione del codice seguente? (Una risposta)

```
import java.io.IOException;

public class ExceptionTest {
    public static void main (String[] args) {
        try {
            methodA();
        }
        catch(IOException e) {
            System.out.println("Eccezione IO");
        }
        catch(Exception e) {
            System.out.println("Eccezione");
        }
    }
}
```

```
public void methodA() {  
    throw new IOException();  
}  
}
```

- A. Il programma viene compilato ed eseguito normalmente, ma non stampa nulla.  
B. Eccezione IO  
C. Eccezione  
D. Errore di compilazione.  
E. Viene lanciata una eccezione in esecuzione.
- 

10. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
1. class Purple {  
2.     public static void main (String []args) {  
3.         int[] i = {1,2,3};  
4.         Object obj = i;  
5.         i = obj;  
6.     }  
7. }
```

- A. Errore di compilazione alla linea 4.  
B. Errore di compilazione alla linea 5.  
C. Errore di esecuzione alla linea 4.  
D. Errore di esecuzione alla linea 5.  
E. Il programma viene compilato ed eseguito senza errori.
- 

11. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
class Black {  
    static byte a;  
    static short b;  
    static char c;  
    static int d;  
    static long e;  
    static String s;  
    public static void main(String[] args) {  
        System.out.println(a+b+c+d+e+s);  
    }  
}
```

- A. 00000null  
B. 0null  
C. 0  
D. null  
E. Errore di compilazione.
- 

12. Quali sono i due sovraccaricamenti corretti dell'unico metodo della classe seguente? (Due risposte)

```
public class A {  
    public void f(int a, float c) {}  
}
```

- A. private void f(int a, int b) {}  
B. protected void f(int a, float b) {}  
C. public int f(int a, float c) {return a;}  
D. public int f(float c, int a) {return a;}  
E. float f(int a, float c) {return c;}
- 

13. Qual è il risultato della compilazione ed esecuzione del codice seguente? (Una risposta)

```
public class Foo {  
    public static void main (String[] args) {  
        String s;  
        System.out.println ("s=" + s);  
    }  
}
```

- A. s=  
B. s=null  
C. Il codice non viene compilato poiché la stringa s non è inizializzata.  
D. Il codice non viene compilato poiché la stringa s non può essere referenziata.  
E. Errore di esecuzione.
- 

14. Qual è il risultato della compilazione ed esecuzione del codice seguente? (Una risposta)

```
public class Foo {  
    public static void main (String[] args) {  
        try {return;}  
        finally {System.out.println("Fine");}  
    }  
}
```

- A. Il programma viene compilato ed eseguito, ma non stampa nulla.  
B. Fine  
C. Il programma viene compilato, ma lancia una eccezione in esecuzione.  
D. Il programma non viene compilato poiché manca il blocco catch.  
E. Il programma non viene compilato per un motivo diverso da quello al punto precedente.
- 

15. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
1. interface I1 {}  
2. interface I2 {}  
3. class Base implements I1 {}  
4. class Sub extends Base implements I2 {}  
5.  
6. class Red {  
7.     public static void main(String args[]) {  
8.         Sub s1 = new Sub();
```

```
9.     I2 i2 = s1;
10.    I1 i1 = s1;
11.    Base base = s1;
12.    Sub s2 = (Sub)base;
13. }
14. }
```

- A. Errore di compilazione alla linea 9 o alla 10.
  - B. Errore di compilazione alla linea 11.
  - C. Errore di compilazione alla linea 12.
  - D. Errore in esecuzione.
  - E. Il programma viene compilato ed eseguito senza errori.
- 
16. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
interface I {
    String s1 = "I";
}

class A implements I {
    String s1 = "A";
}

class B extends A {
    String s1 = "B";
}

class C extends B {
    String s1 = "C";
    void printIt() {
        System.out.print(((A)this).s1 +
                        ((B)this).s1 +
                        ((C)this).s1 +
                        ((I)this).s1);
    }
    public static void main (String[] args) {
        new C().printIt();
    }
}
```

- A. ABCI
  - B. Errore di esecuzione.
  - C. Errore di compilazione, poiché una interfaccia non può contenere variabili.
  - D. Errore di compilazione a causa di un cast illegale.
  - E. Un risultato diverso dai precedenti.
-

**LP1 – Prova d'esame****Foglio delle risposte**

proff. Eliana Minicozzi e Marcello Sette

13 settembre 2007

Studente e matricola:

Gruppo [1:Minicozzi, 2:Sette]:

|    |   |   |   |   |   |
|----|---|---|---|---|---|
| 1  | A | B | C | D | E |
| 2  | A | B | C | D | E |
| 3  | A | B | C | D | E |
| 4  | A | B | C | D | E |
| 5  | A | B | C | D | E |
| 6  | A | B | C | D | E |
| 7  | A | B | C | D | E |
| 8  | A | B | C | D | E |
| 9  | A | B | C | D | E |
| 10 | A | B | C | D | E |
| 11 | A | B | C | D | E |
| 12 | A | B | C | D | E |
| 13 | A | B | C | D | E |
| 14 | A | B | C | D | E |
| 15 | A | B | C | D | E |
| 16 | A | B | C | D | E |

Valutazione della prova 1:

Segnare nell'ultimo foglio le risposte alle seguenti domande a scelta multipla, tenendo presente che alcune domande potrebbero richiedere una risposta multipla.

1. Qual è il risultato della compilazione ed esecuzione del programma seguente? (Una risposta)

```

1. class A {
2.     public int getN(int a) {
3.         return a+1;
4.     }
5. }
6.
7. class B extends A {
8.     public int getN(int a) {
9.         return a+2;
10.    }
11.   public static void main(String args[]) {
12.       A a = new B();
13.       System.out.println(a.getN(0));
14.   }
15. }
```

- A. 1
- B. 2
- C. Errore di compilazione alla linea 8.
- D. Errore di compilazione alla linea 12.
- E. Errore di compilazione alla linea 13.

2. In una classe pubblica, quale modificatore di accesso bisogna usare per consentire solo ai membri della classe l'accesso ad un metodo della classe? (Una risposta)

- A. `public`
- B. `private`
- C. `protected`
- D. `default`
- E. Nessun modificatore di accesso.

3. Quali tre modifiche (fatte alternativamente ed indipendentemente) rendono il seguente codice compilabile? (Tre risposte)

```

1. public abstract class Test {
2.     public abstract void methodA();
3.
4.     public abstract void methodB()
5.     {
6.         System.out.println("Ciao");
7.     }
8. }
```

- A. Aggiungere il corpo a `methodA`.
- B. Sostituire le linee da 5 a 7 con un punto e virgola (“;”).
- C. Rimuovere il qualificatore `abstract` dalla dichiarazione di `Test`.
- D. Rimuovere il qualificatore `abstract` dalla dichiarazione di `methodB`.
- E. Rimuovere interamente `methodB` e sostituire la parola `class` con la parola `interface` alla linea 1.

4. Dato:

```

1. public interface Foo {
2.     int k = 4;
3. }
```

Quali tre linee sono equivalenti alla linea 2? (Tre risposte)

- A. `final int k = 4;`
- B. `public int k = 4;`
- C. `static int k = 4;`
- D. `private int k = 4;`
- E. `abstract int k = 4;`

5. Quale brano può essere aggiunto al punto X per rendere compilabile il seguente codice? (Una risposta)

```
public class ExceptionTest {
    class TestException extends Exception {}
```

```
public void runTest() throws TestException {}
```

```
public void test() /* Punto X */ {
    runTest();
}
```

- A. `throws Exception`
- B. `catch(Exception e)`
- C. `throws RuntimeException`
- D. `catch(TestException e`
- E. Non è necessaria nessuna modifica.

6. Quando l'oggetto creato alla linea 4 diventa eleggibile per la garbage collection? (Una risposta)

```

1. public class X {
2.     public Object m() {
3.         Object o = new String("Cucu");
4.         Object [] oa = new Object [1];
5.         oa[0] = o;
6.         o = null;
7.         oa[0] = null;
8.         return o;
9.     }
10. }
```

- A. Dopo la linea 5.
- B. Dopo la linea 6.

- C. Dopo la linea 7.
  - D. Dopo la linea 8 (alla terminazione del metodo).
  - E. Non si può dire, poiché l'oggetto viene restituito dal metodo ad un oggetto del quale non conosciamo il ciclo di vita.
- 

7. Dato un file contenente:

```
1. public class MioCerchio {  
2.     public double raggio;  
3.     public double diametro;  
4.     public void setRaggio(double raggio) {  
5.         this.raggio = raggio;  
6.         this.diametro = raggio * 2;  
7.     }  
8.     public double getRaggio() {  
9.         return raggio;  
10.    }  
11. }
```

Quale delle seguenti affermazioni è vera? (Una risposta)

- A. La classe **MioCerchio** è ben encapsulata.
  - B. In ogni istanza di **MioCerchio**, il **diametro** sarà sempre il doppio del **raggio**.
  - C. Le linee 5 e 6 sono sufficienti a soddisfare i requisiti dell'incapsulamento.
  - D. In ogni istanza di **MioCerchio**, il **raggio** può essere modificato indipendentemente dal **diametro**.
  - E. Nessuna delle precedenti è vera.
- 

8. Quali delle seguenti affermazioni sono vere? (Due risposte)

- A. Il costruttore di default inizializza le variabili dei metodi.
  - B. Il compilatore crea sempre un costruttore di default per ogni classe.
  - C. Il costruttore di default di solito invoca il costruttore senza parametri della superclasse.
  - D. Il costruttore di default inizializza le variabili di istanza dichiarate nella classe.
  - E. Quando in una classe è definito un solo costruttore con parametri, il compilatore non fornisce il costruttore di default.
- 

9. Quale affermazione riguardante il brano di codice seguente è vera? (Una risposta)

```
int index = 1;  
String [] test = new String[3];  
String foo = test[index];
```

- A. **foo** ha valore "".
  - B. **foo** ha valore **null**.
  - C. Errore in esecuzione.
  - D. Errore di compilazione.
  - E. Nessuna delle precedenti è vera.
- 

10. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
1. public class Test {  
2.     public static void main(String []args) {  
3.         class Foo {  
4.             public int i = 3;  
5.         }  
6.         Object o = (Object) new Foo();  
7.         Foo foo = (Foo) o;  
8.         System.out.println(foo.i);  
9.     }  
10. }
```

- A. Errore di compilazione alla linea 6.
  - B. Errore di compilazione alla linea 7.
  - C. Errore di esecuzione alla linea 6.
  - D. Errore di esecuzione alla linea 7.
  - E. 3
- 

11. Dato:

```
public class Test {  
    private int x, y;  
    private float z;  
    public void setVar(int a, int b, float c) {  
        x = a;  
        y = b;  
        z = c;  
    }  
}
```

Quali due metodi sono dei sovraccaricamenti (overload) del metodo **setVar**? (Due risposte)

- A. void **setVar**(int a, int b, float c) {  
 x = a;  
 y = b;  
 z = c;  
}
  - B. public void **setVar**(int a, float c, int b) {  
 setVar(a, b, c);  
}
  - C. public void **setVar**(int a, float c, int b) {  
 this(a, b, c);  
}
  - D. public void **setVar**(int a, float b) {  
 x = a;  
 z = b;  
}
  - E. public void **setVar**(int ax, int by, float cz) {  
 x = ax;  
 y = by;  
 z = cz;  
}
- 

12. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
1. class A {  
2.     public byte getN() {  
3.         return 1;  
4.     }  
5. }  
6.
```

```

7. class B extends A {
8.     public short getN() {
9.         return 2;
10.    }
11.    public static void main(String args[]) {
12.        B b = new B();
13.        System.out.println(b.getN());
14.    }
15. }

```

- A. 1
- B. 2
- C. Errore di compilazione alla linea 8.
- D. Errore di compilazione alla linea 13.
- E. Errore di esecuzione alla linea 13.

13. Qual è il risultato della compilazione ed esecuzione dei file seguenti? (Una risposta)

**La risposta è A  
e non B**

**ClassOne.java**

```

1. package com.abc.pkg1;
2. public class ClassOne {
3.     private char var = 'a';
4.     chargetVar() { return var; }
5. }

```

**ClassTest.java**

```

1. package com.abc.pkg2;
2. import com.abc.pkg1.ClassOne;
3. public class ClassTest extends ClassOne {
4.     public static void main(String[] args) {
5.         char a = new ClassOne().getVar();
6.         char b = new ClassTest().getVar();
7.     }
8. }

```



- A. Errore di compilazione.
- B. Nessun errore di compilazione e di esecuzione.
- C. Nessun errore di compilazione, ma errore in esecuzione alla linea 5 in **ClassTest.java**.
- D. Nessun errore di compilazione, ma errore in esecuzione alla linea 6 in **ClassTest.java**.
- E. Nessuna delle precedenti è vera.

14. Dato il codice seguente:

```

public class Test {
    public static void main(String[] args) {
        String foo = args[1];
        String bar = args[2];
        String baz = args[3];
        System.out.println("baz = " + baz);
    }
}

```

Quale delle seguenti linee di comando produrranno l'output:

**baz = 2**

- A. java Test 2222
- B. java Test 1 2 3 4

- C. java Test 4 4 4 2
- D. java Test 4 3 2 1
- E. java Test 2 3 4 1

15. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```

1. public class Foo {
2.     String s;
3.     public static void main(String[] args) {
4.         System.out.println(s);
5.     }
6. }

```

- A. Errore di compilazione alla linea 2.
- B. Errore di compilazione alla linea 4.
- C. null
- D. Errore di esecuzione.
- E. Il programma viene compilato ed eseguito correttamente ma non stampa nulla.

16. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```

1. class A {
2.     public String toString() {
3.         return "4";
4.     }
5. }
6.
7. class B extends A {
8.     public String toString() {
9.         return super.toString() + "3";
10.    }
11. }
12.
13. public class Test {
14.     public static void main(String[] args) {
15.         System.out.println(new B());
16.     }
17. }

```

- A. 4
- B. 43
- C. Errore di compilazione alla linea 9.
- D. Errore di compilazione alla linea 15.
- E. Errore di esecuzione.

**LP1 – Prova d'esame****Foglio delle risposte**

proff. Eliana Minicozzi e Marcello Sette

1 dicembre 2007

Studente e matricola:

Gruppo [1:Minicozzi, 2:Sette]:

|    |   |   |   |   |   |
|----|---|---|---|---|---|
| 1  | A | B | C | D | E |
| 2  | A | B | C | D | E |
| 3  | A | B | C | D | E |
| 4  | A | B | C | D | E |
| 5  | A | B | C | D | E |
| 6  | A | B | C | D | E |
| 7  | A | B | C | D | E |
| 8  | A | B | C | D | E |
| 9  | A | B | C | D | E |
| 10 | A | B | C | D | E |
| 11 | A | B | C | D | E |
| 12 | A | B | C | D | E |
| 13 | A | B | C | D | E |
| 14 | A | B | C | D | E |
| 15 | A | B | C | D | E |
| 16 | A | B | C | D | E |

Valutazione della prova 1:

# LP1 – Prova Intermedia 1

prof. Eliana Minicozzi

prof. Marcello Sette

21 aprile 2007

## Esercizio 1 – max 7/30

La procedura `new(...)` in Pascal e la funzione `malloc(...)` in C sono utilizzate per allocare spazio di memoria sullo heap; lo stesso spazio viene liberato attraverso l'invocazione di `dispose(...)` e di `free(...)`, rispettivamente.

Discutere, aiutandosi con alcuni diagrammi, quello che succede ai Data Object e ai legami durante l'esecuzione di ogni singola linea di uno a scelta dei seguenti programmi:

### Versione Pascal

```
program esercizio1(input, output);
var
  a: integer;
  p, q: ^integer;
begin
  a:= 7;
  new(p);
  q:= p;
  q^:= a;
  writeln(p^);
  dispose(p)
end.
```

### Versione C

```
main() {
  int a = 7;
  int *p, *q;
  p = (int *) malloc (sizeof(int));
  q = p;
  *q = a;
  printf("%d", *p);
  free(p);
}
```

## Esercizio 2 – max 8/30

Sapendo che, nel linguaggio in uso, i parametri `var` delle procedure sono parametri INOUT realizzati per riferimento, quelli non-`var` sono parametri IN realizzati per copia, aiutandosi con un diagramma in cui rappresentare lo stack di attivazione, discutere il comportamento del successivo programma ed il suo output

1. nel caso in cui il linguaggio usi ambito statico di validità dei legami;
2. nel caso in cui il linguaggio usi ambito dinamico di validità dei legami.

```
program esercizio2 (input, output);
var
  a, b: integer;

procedure p1(var b: integer);
begin
  a:= a+b;  b:= b+1
end;

procedure p2(var a: integer);
  procedure p3(a: integer);
  begin
    b:= a*2;
    p1(a);
    writeln(a, b)
  end;
begin
  b:= a+1;
  p3(b);
  a:= b-1;
  writeln(a, b)
end;

begin
  a:= 1;  b:= 2;
  p2(b);
  writeln(a, b)
end.
```

## Esercizio 3

### Prima parte – max 7/30

Si vuole costruire un semplice modello a oggetti che rappresenti i personaggi dei cartoni animati per simulare solo la loro capacità di movimento.

In questo momento della progettazione siamo in grado di dire solo che:

- ogni personaggio ha un solo tipo di movimento;
- sono noti due tipi di movimento: con le ali, oppure con le zampe;
- sono noti due tipi di personaggi: gli uccelli e i mammiferi.

Dato il seguente diagramma dei casi d'uso:



descrivere il successivo diagramma di sequenza, specificando:

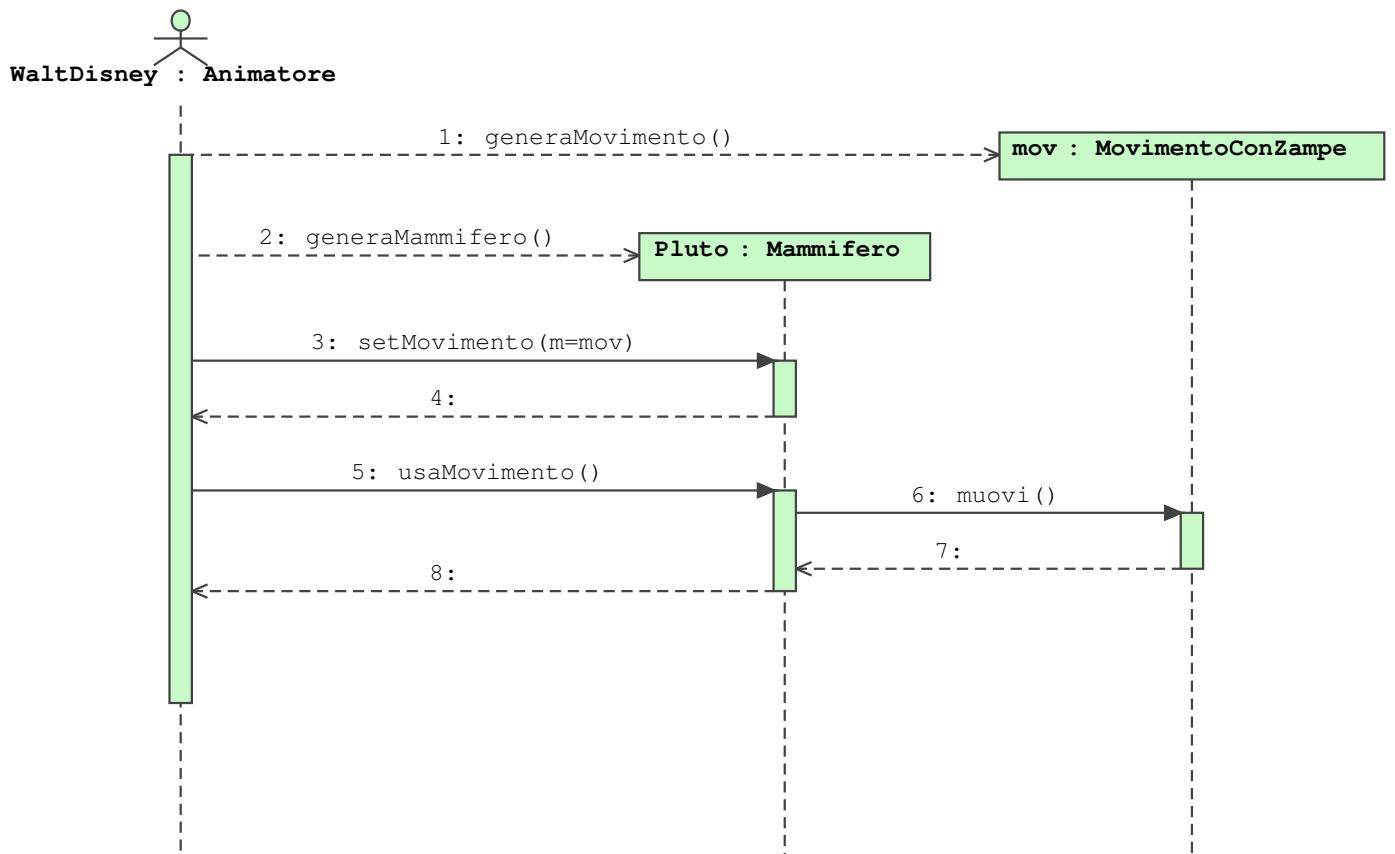
1. il significato dei simboli grafici presenti;
2. a chi appartengono i metodi `setMovimento`, `usaMovimento`, `muovi`;

3. il nome del caso d'uso che si sta sceneggiando;
4. la narrazione dello scenario rappresentato.

### Seconda parte – max 8/30

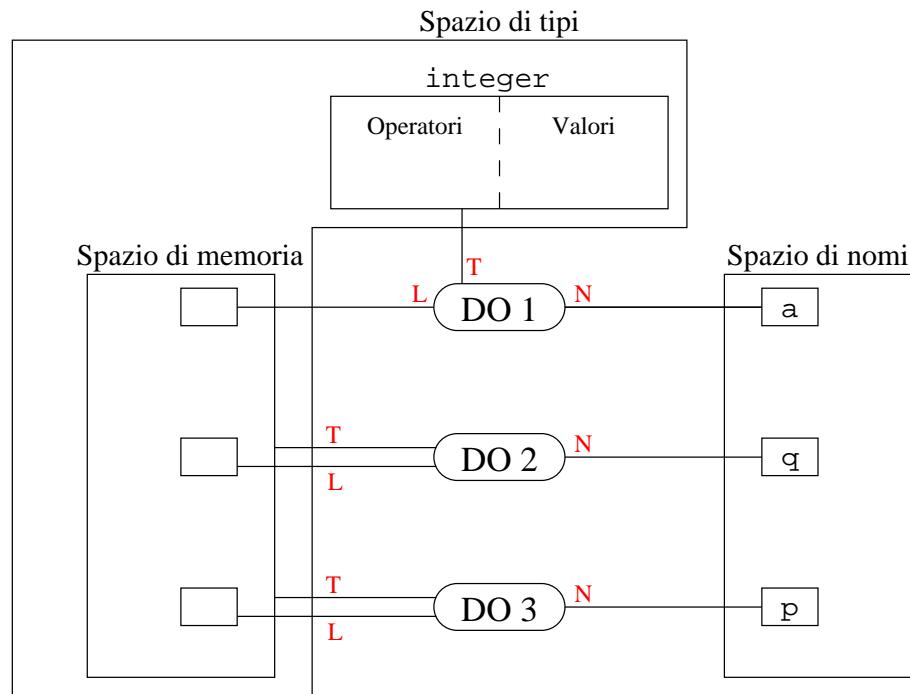
1. Costruire un diagramma di classe compatibile con la prima parte dell'esercizio.
2. Costruire un diagramma degli oggetti contenente l'uccellino Titti e il cane (mammifero) Pluto con i loro rispettivi movimenti.
3. Aggiungere al diagramma degli oggetti Paperino: esso è un uccello il cui movimento è con le zampe e non con le ali.
4. Si vuole aggiungere ancora un nuovo oggetto: il delfino Flipper. Ma non c'è il movimento adatto a Flipper: esso è un mammifero che si muove con le pinne.

Modificare il diagramma di classe prodotto al punto 1, in modo che esso consenta la generazione anche di Flipper.

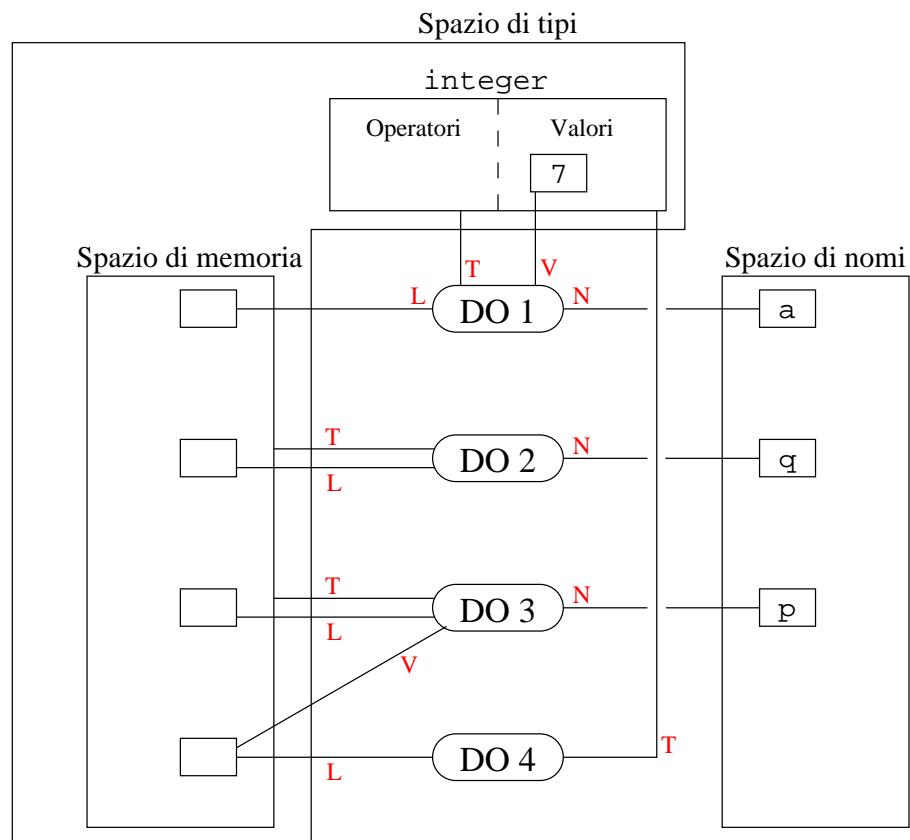


## Soluzione esercizio 1

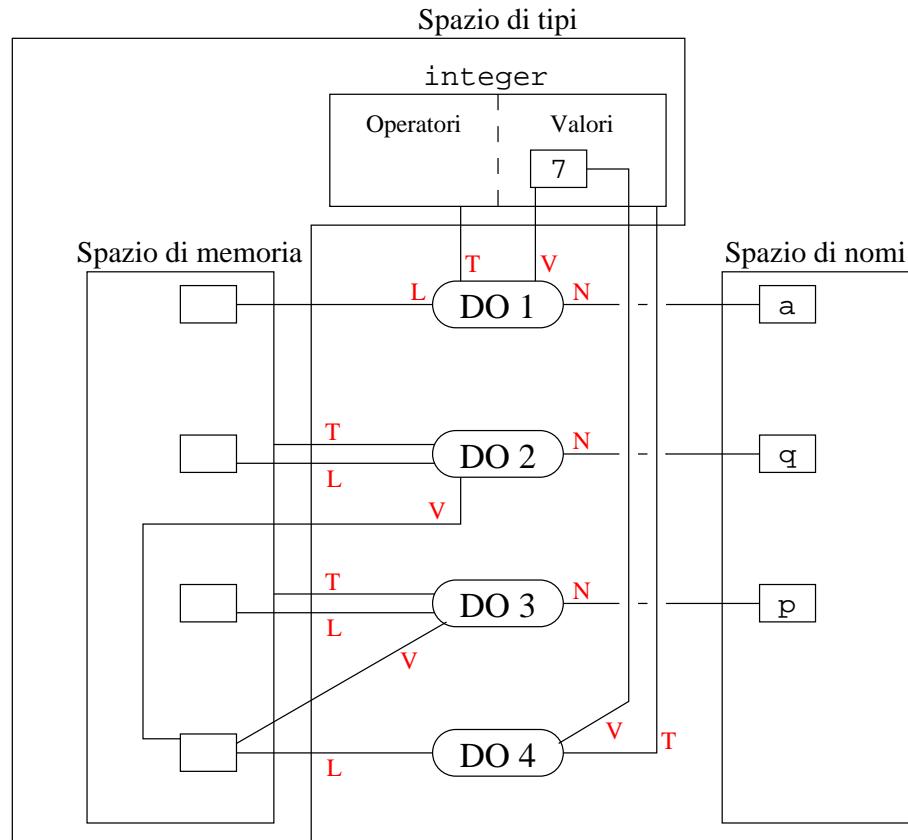
Analizzando il programma in Pascal (l'altro è identico, tranne per l'inizializzazione della variabile `a` che avviene nel momento della dichiarazione), prima dell'esecuzione dell'istruzione `a := 7; :`



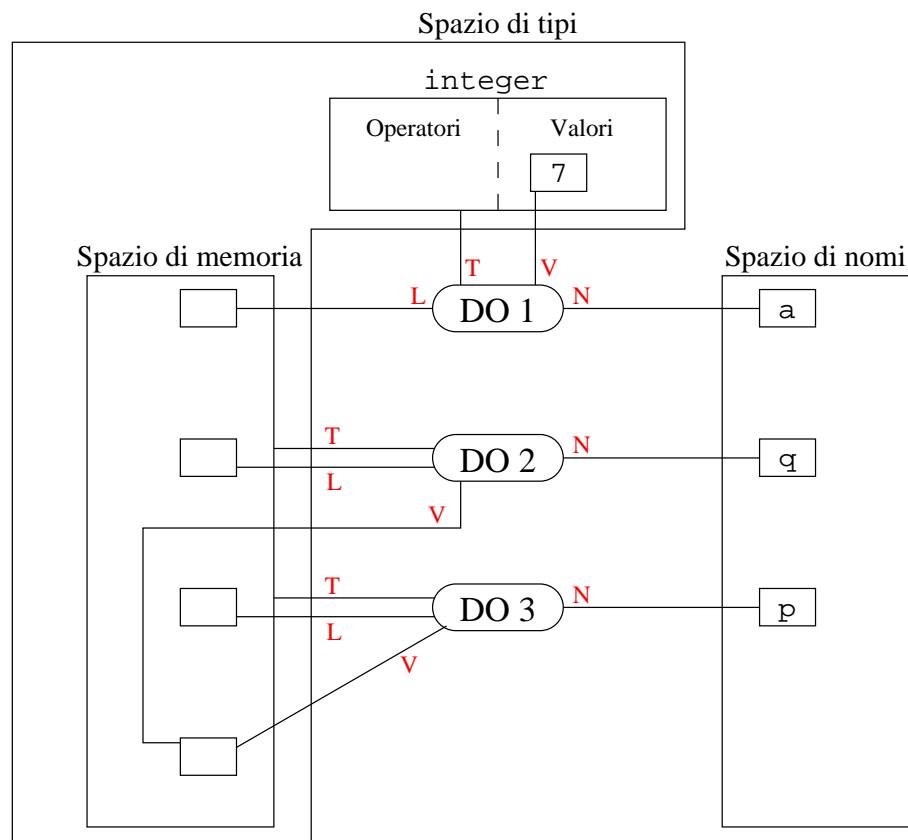
Prima dell'esecuzione dell'istruzione `q := p; :`



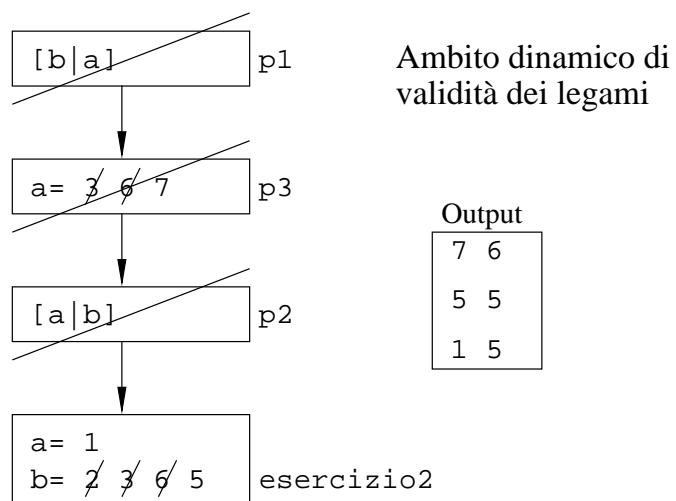
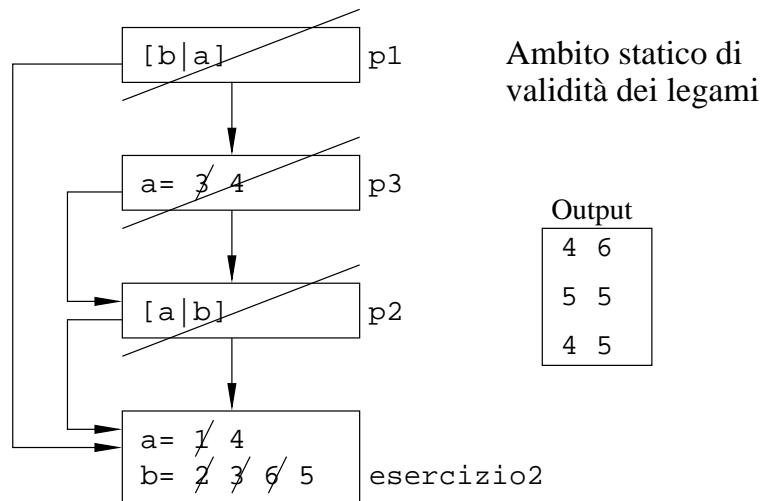
Le due istruzioni `q := p`; e `q^ := a`; causano l'instaurazione di due legami di valore:



Infine, dopo l'esecuzione di `dispose(p)` :



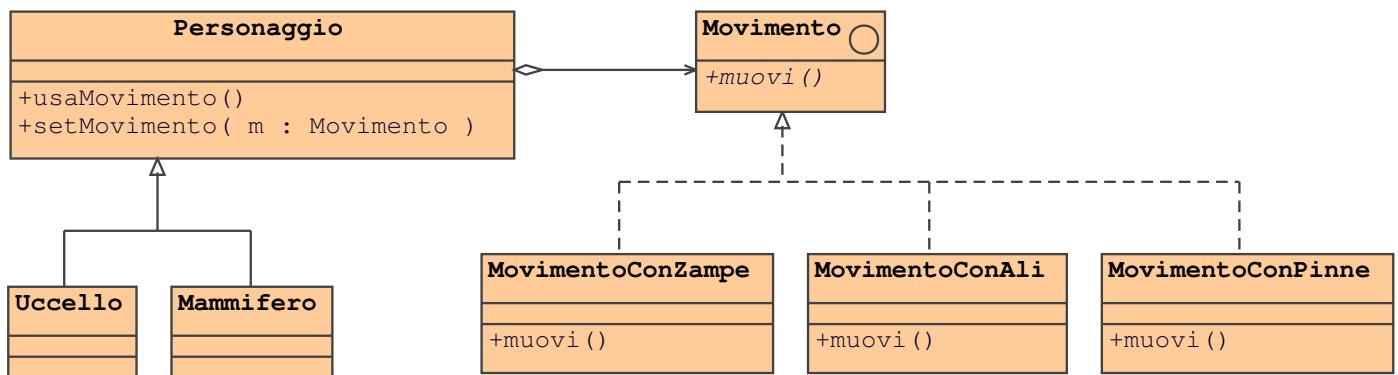
## Soluzione esercizio 2



Legenda:

- $[a|b]$ : parametro formale a, parametro attuale b, passaggio INOUT per riferimento.
- $a=4$ : parametro formale a, valore del parametro attuale 4, passaggio IN per copia.

## Soluzione esercizio 3



## LP1

## Prova d'esame

proff. E. Minicozzi e M. Sette

11 gennaio 2008

Segnare nell'ultimo foglio le risposte alle seguenti domande a scelta multipla, tenendo presente che alcune domande potrebbero richiedere una risposta multipla.

1. Qual è il risultato della compilazione ed esecuzione del programma seguente? (Una risposta)

```
class A {
    public static void main(String[] args) {
        int[] a1 = {1,2,3};
        int []a2 = {4,5,6};
        int a3[] = {7,8,9};
        System.out.print(
            a1[1]+","+a2[1]+","+a3[1]);
    }
}
```

- A. Stampa: 1,4,7
- B. Stampa: 2,5,8
- C. Stampa: 3,6,9
- D. Errore di compilazione.
- E. Errore di esecuzione.

2. Quali delle seguenti affermazioni sono vere? (Tre risposte)

- A. L'incapsulazione serve a nascondere i dati contenuti.
- B. Una classe strettamente encapsulata è sempre immutabile.
- C. L'incapsulazione è sempre usata per rendere i programmi più veloci.
- D. L'incapsulazione serve a proteggere i dati da corruzione.
- E. L'incapsulazione permette modifiche del disegno interno di una classe, conservando l'aspetto dell'interfaccia pubblica.

3. Quali oggetti appartengono all'insieme degli oggetti elegibili per la garbage collection alla terminazione del metodo m1? (Numero di risposte indeterminato)

```
class I {
    private I other;
    public void other(I i) {other = i;}
}

class J {
    private void m1() {
        I i1 = new I(), i2 = new I();
        I i3 = new I(), i4 = new I();
        i1.other(i3); i2.other(i1);
        i3.other(i2); i4.other(i4);
    }
}
```

```
}
public static void main (String[] args) {
    new J().m1();
}
A. i1
B. i2
C. i3
D. i4
E. Nessuno dei precedenti.
```

4. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
class A {
    public static void main(String[] args) {
        int x, y, z;
        System.out.println(x+y+z);
    }
}
```

- A. Non stampa nulla.
- B. Stampa: null
- C. Stampa: 0
- D. Errore di compilazione.
- E. Errore di esecuzione.

5. Quali dei seguenti modificatori non possono essere usati con `abstract` nella dichiarazione di un metodo? (Tre risposte)

- A. `final`
- B. `private`
- C. `protected`
- D. `public`
- E. `static`

6. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
class L1Exception extends Exception {}
class L2Exception extends L1Exception {}
class L3Exception extends L2Exception {}
```

```
class P {
    public static void main(String[] args) {
        int a,b,c,d,f,g,x;
        a = b = c = d = f = g = 0;
        x = 3;
        try {
            try {
                switch (x) {
                    case 1: throw new L1Exception();
                    case 2: throw new L2Exception();
                    case 3: throw new L3Exception();
                }
                a++;
            }
            catch (L2Exception e) {b++;}
            finally {c++;}
        }
    }
}
```

```

    }
    catch (L1Exception e) {d++;}
    catch (Exception e) {f++;}
    finally {g++;}
    System.out.print(a + "," + b + "," + c +
                     "," + d + "," + f + "," + g);
}
}

```

- A. Stampa: 1,1,1,0,0,1
- B. Stampa: 0,1,1,0,0,1
- C. Stampa: 0,1,0,0,0,0
- D. Stampa: 0,1,0,0,0,1
- E. Stampa: 0,0,1,0,0,1

7. Due delle affermazioni riguardanti il codice seguente sono vere. Quali? (Due risposte)

```

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

```

- A. Il compilatore cerca di creare un costruttore di default per la classe A.
- B. Il compilatore cerca di creare un costruttore di default per la classe B.
- C. Errore di compilazione alla linea 1.
- D. Errore di compilazione alla linea 2.
- E. Una sola delle precedenti è vera.

8. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```

class A {
    public static void main (String[] args) {
        private int x = 1;
        protected int y = 2;
        public int z = 3;
        System.out.println(x+y+z);
    }
}

```

- A. Stampa: 123
- B. Stampa: 1 2 3
- C. Stampa: 6
- D. Errore di compilazione.
- E. Errore di esecuzione.

9. Qual è il risultato della compilazione del seguente programma? (Una risposta)

```

class A {void m1(String s1) {}}
class B extends A {
    void m1(String s1) {} // 1
    void m1(boolean b) {} // 2
    void m1(byte b) throws Exception {} // 3
    String m1(short s) {return "";} // 4
    private void m1(char c) {} // 5
}

```

- A. Errore di compilazione alla riga 1 o alla riga 2.
- B. Errore di compilazione alla riga 3.

C. Errore di compilazione alla riga 4.

D. Errore di compilazione alla riga 5.

E. Nessun errore.

10. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```

class A {
    static String m(float i) {return "float";}
    static String m(double i) {
        return "double";
    }
    public static void main (String[] args) {
        int a = 1;
        long b = 2;
        System.out.print(m(a) + "," + m(b));
    }
}

```

- A. Stampa: float,float
- B. Stampa: float,double
- C. Stampa: double,float
- D. Stampa: double,double
- E. Errore di compilazione o di esecuzione.

11. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```

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

```

- A. Stampa: 1,1
- B. Stampa: 1,2
- C. Stampa: 2,1
- D. Stampa: 2,2
- E. Errore di compilazione o esecuzione.

12. Le definizioni successive sono contenute in uno stesso file di nome B.java (pertanto non classi annidate, ma classi top-level). A quali linee sono generati errori di compilazione? (Tre risposte)

```

public class B {} // 1
class B1 {} // 2
protected class B2 {} // 3
private class B3 {} // 4
Class B4 {} // 5

```

- A. 1
- B. 2
- C. 3
- D. 4
- E. 5

13. Qual è il risultato della compilazione ed esecuzione del programma seguente? (Una risposta)

```
class A {  
    public static void main (String[] args) {  
        Error error = new Error();  
        Exception exception = new Exception();  
        System.out.print(  
            (error instanceof Throwable) + "," );  
        System.out.print(  
            (exception instanceof Throwable));  
    }  
}
```

- A. Errore di compilazione.
- B. Stampa: **false, false**
- C. Stampa: **false, true**
- D. Stampa: **true, false**
- E. Stampa: **true, true**

14. Quali delle seguenti affermazioni sono vere? (Due risposte)

- A. Un costruttore può invocare un altro costruttore della stessa classe usando “this(argumentList<sub>opt</sub>);”
- B. Un costruttore può invocare se stesso usando “this(argumentList<sub>opt</sub>);”
- C. L’invocazione “this(argumentList<sub>opt</sub>);” può apparire dovunque nel corpo di un costruttore.
- D. Un costruttore può invocare un costruttore della superclasse diretta usando “super(argumentList<sub>opt</sub>);”
- E. Il numero di invocazioni di costruttori che possono essere presenti nel corpo di uno stesso costruttore non può eccedere il numero di costruttori totalmente definiti nella stessa classe.

15. A quali linee vengono generati errori di compilazione? (Numero di risposte indeterminato)

```
interface A {  
    void m1();           // 1  
    public void m2();    // 2  
    protected void m3(); // 3  
    private void m4();   // 4  
}
```

- A. Linea 1.
- B. Linea 2.
- C. Linea 3.
- D. Linea 4.
- E. Nessun errore di compilazione.

16. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
class A {  
    void m(A a) {System.out.print("A");}  
}  
class B extends A {  
    void m(B b) {System.out.print("B");}  
}
```

```
class C extends B {  
    void m(C c) {System.out.print("C");}  
}  
class D extends C {  
    void m(D d) {System.out.print("D");}  
    public static void main(String[] args) {  
        A a = new A(); B b = new B();  
        C c = new C(); D d = new D();  
        d.m(a); d.m(b); d.m(c);  
    }  
}
```

- A. Stampa: AAA
- B. Stampa: ABC
- C. Stampa: DDD
- D. Errore di compilazione.
- E. Errore di esecuzione.

**LP1 – Prova d'esame****Foglio delle risposte**

proff. Eliana Minicozzi e Marcello Sette

11 gennaio 2008

Studente e matricola:

Gruppo [1:Minicozzi, 2:Sette]:

|    |   |   |   |   |   |
|----|---|---|---|---|---|
| 1  | A | B | C | D | E |
| 2  | A | B | C | D | E |
| 3  | A | B | C | D | E |
| 4  | A | B | C | D | E |
| 5  | A | B | C | D | E |
| 6  | A | B | C | D | E |
| 7  | A | B | C | D | E |
| 8  | A | B | C | D | E |
| 9  | A | B | C | D | E |
| 10 | A | B | C | D | E |
| 11 | A | B | C | D | E |
| 12 | A | B | C | D | E |
| 13 | A | B | C | D | E |
| 14 | A | B | C | D | E |
| 15 | A | B | C | D | E |
| 16 | A | B | C | D | E |

Valutazione della prova 1:

Segnare nell'ultimo foglio le risposte alle seguenti domande a scelta multipla, tenendo presente che alcune domande potrebbero richiedere una risposta multipla.

1. Qual è il risultato della compilazione ed esecuzione del programma seguente? (Una risposta)

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

- A. Stampa: 1,3
- B. Stampa: 2,3
- C. Stampa: 1,4
- D. Errore di compilazione.
- E. Nessuna delle precedenti.

2. Quale delle seguenti tecniche può essere usata per impedire l'istanziazione di una classe da parte di codice posto all'esterno della classe? (Una risposta)

- A. Non dichiarare nessun costruttore.
- B. Non usare una istruzione `return` nel costruttore.
- C. Dichiarare tutti i costruttori usando la parola chiave `void` per indicare che non viene ritornato nulla.
- D. Definire almeno un costruttore e dichiarare tutti i costruttori definiti usando il modificatore di accesso `private`.
- E. Nessuna delle precedenti.

3. Quali dei tre oggetti A, B o C non è eleggibile per la garbage collection quando il metodo `m2` comincia ad essere eseguito? (Numero di risposte indeterminato)

```
class I {
    private String name;
    public I(String s) {name = s;}
    private I other;
    public void other(I i) {other = i;}
}

class J {
    private I i1 = new I("A");
    private I i2 = new I("B");
    private I i3 = new I("C");

    private void m1() {
```

```
i1.other(i2); i2.other(i1); i3.other(i3);
i1 = i3; i2 = i3;
m2();
}

private void m2() {/* Non si sa cosa fa */}

public static void main (String[] args) {
    new J().m1();
}
}

A. A
B. B
C. C
D. Nessuno dei precedenti.
E. Errore di compilazione.
```

4. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
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();
    }
}
```

- A. Stampa: `main,B.m1`
- B. Stampa: `main,A.m1`
- C. Stampa: `A.m1`
- D. Errore di compilazione.
- E. Errore di esecuzione.

5. Quale delle seguenti affermazioni è falsa? (Una risposta)

- A. È possibile che una interfaccia sia dichiarata nel corpo di una classe o di una interfaccia.
- B. Una costante può essere membro di una interfaccia.
- C. Una classe può essere membro di una interfaccia.
- D. Se una classe implementa una interfaccia, allora deve implementare tutti i metodi dichiarati nell'interfaccia.
- E. Nessuna delle precedenti è vera.

6. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```

class L1Exception extends Exception {}
class L2Exception extends L1Exception {}
class L3Exception extends L2Exception {}

class P {
    public static void main(String[] args) {
        int a,b,c,d,f,g,x;
        a = b = c = d = f = g = 0;
        x = 4;
        try {
            try {
                switch (x) {
                    case 1: throw new L1Exception();
                    case 2: throw new L2Exception();
                    case 3: throw new L3Exception();
                    case 4: throw new Exception();
                }
                a++;
            }
            catch (L2Exception e) {b++;}
            finally {c++;}
        }
        catch (L1Exception e) {d++;}
        catch (Exception e) {f++;}
        finally {g++;}
        System.out.print(a + "," + b + "," + c +
                         "," + d + "," + f + "," + g);
    }
}

```

- A. Stampa: 0,0,0,0,1,0
  - B. Stampa: 0,0,1,0,0,1
  - C. Stampa: 0,0,1,0,1,1
  - D. Stampa: 0,1,1,1,1,1
  - E. Stampa: 1,1,1,1,1,1
- 

7. Quali dei seguenti modificatori può essere applicato ad una classe che non è interna? (Tre risposte)

- A. **public**
  - B. **static**
  - C. **private**
  - D. **abstract**
  - E. **final**
- 

8. Qual è il risultato della compilazione ed esecuzione del codice seguente? (Una risposta)

```

class A {
    public static void main (String[] args) {
        Object e = new Error();
        Object r = new RuntimeException();
        System.out.print((e instanceof Exception) +
                         System.out.print(r instanceof Exception);
    }
}

```

- A. Stampa: **false**,**false**
  - B. Stampa: **false**,**true**
  - C. Stampa: **true**,**false**
  - D. Stampa: **true**,**true**
- 

E. Errore di compilazione o di esecuzione.

---

9. Qual è il risultato della compilazione del seguente programma? (Una risposta)

```

class A {
    private static int x;
    protected static int y;
    public static int z;
    public static void main (String[] args) {
        System.out.println(x+y+z);
    }
}

```

- A. Non stampa nulla.
  - B. Stampa: **null**
  - C. Stampa: 0
  - D. Errore di compilazione.
  - E. Errore di esecuzione.
- 

10. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```

class A {
    static String m(float i) {return "float";}
    static String m(double i) {
        return "double";
    }
    public static void main (String[] args) {
        char a = 1;
        long b = 2;
        System.out.print(m(a) + "," + m(b));
    }
}

```

- A. Stampa: **float**,**float**
  - B. Stampa: **float**,**double**
  - C. Stampa: **double**,**float**
  - D. Stampa: **double**,**double**
  - E. Errore di compilazione o di esecuzione.
- 

11. Quale delle seguenti affermazioni è vera? (Una risposta)

- A. Un metodo astratto non può essere sovrapposto da un metodo astratto.
  - B. Un metodo di istanza che non è astratto non può essere sovrapposto da un metodo astratto.
  - C. Una dichiarazione di metodo astratto non può includere la clausola **throws**.
  - D. Il corpo di un metodo astratto è rappresentato da parentesi graffe vuote.
  - E. Nessuna delle precedenti.
- 

12. A quali linee vengono generati errori di compilazione? (Due risposte)

```

class A {
    public static void main(String[] args) {
        int[] a1 = new int[];           // 1
        int a2[] = new int[5];          // 2
        int[] a3 = new int[]{1, 2};      // 3
    }
}

```

```

int []a4 = {1, 2};           // 4
int[] a5 = new int[5]{1, 2, 3}; // 5
}

```

---

- A. 1
- B. 2
- C. 3
- D. 4
- E. 5

16. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```

class A {}
class B extends A {}
class C extends B {}

class D {
    void m(A a) {System.out.print("A");}
    void m(B b) {System.out.print("B");}
    void m(C c) {System.out.print("C");}
    public static void main(String[] args) {
        A c1 = new C();
        B c2 = new C();
        C c3 = new C();
        D d = new D();
        d.m(c1); d.m(c2); d.m(c3);
    }
}

```

- A. Stampa: AAA
  - B. Stampa: ABC
  - C. Stampa: DDD
  - D. Errore di compilazione.
  - E. Errore di esecuzione.
- 

13. Quali delle seguenti affermazioni sono vere? (Due risposte)

- A. Una classe top-level non può essere chiamata “strettamente encapsulata” a meno che non sia dichiarata **private**.
- B. L’incapsulazione migliora la manutenibilità del codice.
- C. Una classe strettamente encapsulata consente l’accesso veloce ai propri campi dichiarandoli **public**.
- D. Una classe strettamente encapsulata permette l’accesso ai propri campi solo attraverso metodi.
- E. L’incapsulazione riduce la dimensione del codice.

14. Supponiamo che in un costruttore di una classe appaia esplicitamente l’invocazione di un costruttore di una superclasse, mediante “super(argumentListOpt). Per evitare errori di compilazione, nella lista degli argomenti dell’invocazione non devono essere presenti: (Tre risposte)

- A. Variabili **static** dichiarate nella classe o in una qualunque superclasse.
- B. Variabili di istanza dichiarate nella classe o in una qualunque superclasse.
- C. Metodi **static** dichiarati nella classe o in una qualunque superclasse.
- D. Metodi di istanza dichiarati nella classe o in una qualunque superclasse.
- E. La parola chiave **this**.

15. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```

class A {
    static int x;           // 1
    public static void main(String[] args) {
        int y;             // 2
        System.out.println("x=" + x); // 3
        System.out.println("y=" + y); // 4
    }
}

```

- A. Errore di compilazione alla linea 1.
- B. Errore di compilazione alla linea 2.
- C. Errore di compilazione alla linea 3.
- D. Errore di compilazione alla linea 4.
- E. Nessuna delle precedenti.

**LP1 – Prova d'esame****Foglio delle risposte**

proff. Eliana Minicozzi e Marcello Sette

19 febbraio 2008

Studente e matricola:

Gruppo [1:Minicozzi, 2:Sette]:

|    |   |   |   |   |   |
|----|---|---|---|---|---|
| 1  | A | B | C | D | E |
| 2  | A | B | C | D | E |
| 3  | A | B | C | D | E |
| 4  | A | B | C | D | E |
| 5  | A | B | C | D | E |
| 6  | A | B | C | D | E |
| 7  | A | B | C | D | E |
| 8  | A | B | C | D | E |
| 9  | A | B | C | D | E |
| 10 | A | B | C | D | E |
| 11 | A | B | C | D | E |
| 12 | A | B | C | D | E |
| 13 | A | B | C | D | E |
| 14 | A | B | C | D | E |
| 15 | A | B | C | D | E |
| 16 | A | B | C | D | E |

Valutazione della prova 1:

Segnare nell'ultimo foglio le risposte alle seguenti domande a scelta multipla, tenendo presente che alcune domande potrebbero richiedere una risposta multipla.

1. Qual è il risultato della compilazione ed esecuzione del programma seguente? (Una risposta)

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

- A. Stampa: 1,3
- B. Stampa: 2,3
- C. Stampa: 1,4
- D. Errore di compilazione.
- E. Nessuna delle precedenti.

2. Quale dei seguenti modificatori può essere applicato ad un costruttore? (Una risposta)

- A. `private`
- B. `abstract`
- C. `final`
- D. `static`
- E. Nessuno delle precedenti.

3. Quando inizia l'esecuzione della linea 3, quanti oggetti del tipo Q che sono stati creati alla linea 1 diventano eleggibili per la garbage collection? (Una risposta)

```
void m1() {
    Q q1 = null;
    for (int i=0; i<10; i++) {
        q1 = new Q();           // 1
        m2(q1);                // 2
    }
    System.out.print("Fine"); // 3
}
```

- A. 0
- B. 1
- C. 9
- D. 10
- E. Non è possibile determinare il loro numero.

4. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
class A {
    byte a;
    short b;
    char c;
    int d;
    long e;
    String s;
    public static void main(String[] args) {
        System.out.println(a+b+c+d+e+s);
    }
}
```

- A. Stampa: 00000null
- B. Stampa: 00000
- C. Stampa: Onull
- D. Stampa: 0
- E. Errore di compilazione o di esecuzione.

5. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
class A {
    static String m(int i) {return "int";}
    static String m(float i) {return "float";}
    public static void main(String[] args) {
        long a1 = 1;
        double b1 = 2;
        System.out.println(m(a1) + "," + m(b1));
    }
}
```

- A. Stampa: float,float
- B. Stampa: int,float
- C. Stampa: double,double
- D. Errore di compilazione.
- E. Errore di esecuzione.

6. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
class L1Exception extends Exception {}
class L2Exception extends L1Exception {}
class L3Exception extends L2Exception {}

class P {
    public static void main(String[] args) {
        int a,b,c,d,f,g,x;
        a = b = c = d = f = g = 0;
        x = 1;
        try {
            try {
                switch (x) {
                    case 1: throw new L1Exception();
                    case 2: throw new L2Exception();
                    case 3: throw new L3Exception();
                }
                a++;
            }
            catch (L2Exception e) {b++;}
        }
    }
}
```

```

        finally {c++;}
    }
    catch (L1Exception e) {d++;}
    catch (Exception e) {f++;}
    finally {g++;}
    System.out.print(a + "," + b + "," + c +
                     "," + d + "," + f + "," + g);
}
}

```

- A. Stampa: 0,0,0,1,0,0  
B. Stampa: 0,0,1,1,0,0  
C. Stampa: 0,0,1,1,0,1  
D. Stampa: 1,0,1,1,0,1  
E. Stampa: 1,1,1,1,0,1

7. Supponiamo che le successive siano dichiarazioni di classe top-level non annidate, e che tutte le dichiarazioni siano contenute in un file di nome **Basics.java**. Quali linee non generano un errore di compilazione? (Una risposta)

```

public class Basics {} // 1
public class Basics2 {} // 2
public class Basics3 {} // 3
public class Basics4 {} // 4

```

- A. 1  
B. 1,2  
C. 1,2,3  
D. 1,2,3,4  
E. Non è possibile la compilazione di nessuna linea.

8. Qual è il risultato della compilazione ed esecuzione del codice seguente? (Una risposta)

```

class A {
    private static int x;
    protected static int y;
    public static int z;
    public static void main (String[] args) {
        System.out.println(x+y+z);
    }
}

```

- A. Non stampa nulla.  
B. Stampa un valore indefinito.  
C. Stampa: **null**  
D. Stampa: 0  
E. Errore di compilazione.

9. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```

class A {
    String s1 = "A.s1";
    String s2 = "A.s2";
}

class B extends A {
    String s1 = "B.s1";
    public static void main (String[] args) {

```

```

        B x = new B();
        A y = (A) x;
        System.out.println(x.s1 + " " + x.s2 +
                           " " + y.s1 + " " + y.s2);
    }
}

```

- A. Stampa: B.s1 A.s2 B.s1 A.s2  
B. Stampa: B.s1 A.s2 A.s1 A.s2  
C. Stampa: A.s1 A.s2 B.s1 A.s2  
D. Stampa: A.s1 A.s2 A.s1 A.s2  
E. Errore di compilazione o di esecuzione.

10. Quale tra le seguenti dichiarazioni non produce un errore di compilazione? (Due risposte)

```

interface A {
    protected int e = 5;           // 1
    private int f = 6;             // 2
    int g = 7;                    // 3
    private static int h = 8;       // 4
    public static final int d = 9; // 5
}

```

- A. Quella alla linea 1.  
B. Quella alla linea 2.  
C. Quella alla linea 3.  
D. Quella alla linea 4.  
E. Quella alla linea 5.

11. Quale delle seguenti affermazioni è falsa? (Una risposta)

- A. Un metodo statico è anche chiamato metodo di classe.  
B. Un metodo di classe non è associato con una particolare istanza di quella classe.  
C. La parola “this” non può essere usata nel corpo di un metodo statico.  
D. La parola “super” può essere usata nel corpo di un metodo statico.  
E. Un metodo che non è statico è anche chiamato metodo di istanza.

12. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```

class A {
    public static void main(String[] args) {
        byte[] a = new byte[1];
        long[] b = new long[1];
        float[] c = new float[1];
        Object[] d = new Object[1];
        System.out.print(a[0] + "," + b[0] +
                        "," + c[0] + "," + d[0]);
    }
}

```

- A. Stampa: 0,0,0,null  
B. Stampa: 0,0,0.0,null  
C. Errore di compilazione.

D. Errore di esecuzione.

E. Nessuna delle precedenti.

- 
13. Quale delle seguenti classi non è “strettamente encapsulata”: (Una risposta)

```
class A {private String name;}

class B {
    private String name;
    private void setName(String name) {
        this.name = name;
    }
    private String getName() {
        return name;
    }
}

class C {
    private String name;
    public void setName(String name) {
        this.name = name;
    }
    public String getName() {
        return name;
    }
}
```

- A. La classe A.  
B. La classe B.  
C. La classe C.  
D. Nessuna classe.  
E. Tutte le classi.
- 

14. Qual è il risultato del seguente programma? (Una risposta)

```
public class X {
    public static void main(String args[]) {
        try {
            cattivo();
            System.out.print("A");
        }
        catch(RuntimeException e) {
            System.out.print("B");
        }
        catch(Exception ex) {
            System.out.print("C");
        }
        finally {
            System.out.print("D");
        }
        System.out.print("E");
    }

    public static void cattivo() {
        throw new Exception();
    }
}
```

- A. CD  
B. CDE

C. ACDE

D. ACD

E. Errore di compilazione.

---

15. Quali delle seguenti affermazioni sono vere? (Due risposte)

- A. Un metodo presente in una interfaccia può dichiarare di lanciare eccezioni.  
B. Un metodo dichiarato in una interfaccia deve avere il corpo rappresentato da parentesi graffe vuote.  
C. Una interfaccia può implementare un'altra interfaccia.  
D. Una classe astratta che implementa una interfaccia deve implementare tutti i metodi dichiarati nell'interfaccia.  
E. La dichiarazione di un metodo astratto può essere un membro di una interfaccia.
- 

16. Qual è il risultato della compilazione ed esecuzione del seguente programma? (Una risposta)

```
class A {
    void m(A a) {System.out.print("A");}
}
class B extends A {
    void m(B b) {System.out.print("B");}
}
class C extends B {
    void m(C c) {System.out.print("C");}
}

class D {
    public static void main(String[] args) {
        A c1 = new C();
        B c2 = new C();
        C c3 = new C();
        C c4 = new C();
        c4.m(c1); c4.m(c2); c4.m(c3);
    }
}
```

- A. Stampa: AAA  
B. Stampa: ABC  
C. Stampa: CCC  
D. Errore di compilazione.  
E. Errore di esecuzione.
-

**LP1 – Prova d'esame****Foglio delle risposte**

proff. Eliana Minicozzi e Marcello Sette

3 marzo 2008

Studente e matricola:

Gruppo [1:Minicozzi, 2:Sette]:

|    |   |   |   |   |   |
|----|---|---|---|---|---|
| 1  | A | B | C | D | E |
| 2  | A | B | C | D | E |
| 3  | A | B | C | D | E |
| 4  | A | B | C | D | E |
| 5  | A | B | C | D | E |
| 6  | A | B | C | D | E |
| 7  | A | B | C | D | E |
| 8  | A | B | C | D | E |
| 9  | A | B | C | D | E |
| 10 | A | B | C | D | E |
| 11 | A | B | C | D | E |
| 12 | A | B | C | D | E |
| 13 | A | B | C | D | E |
| 14 | A | B | C | D | E |
| 15 | A | B | C | D | E |
| 16 | A | B | C | D | E |

Valutazione della prova 1:

# LP1 – Prova Intermedia 1

prof. Eliana Minicozzi

prof. Marcello Sette

19 aprile 2008

## Esercizio 1 – max 15/30

Sapendo che, nel linguaggio in uso, la politica di assegnazione dell'ambiente non locale è lo “static scoping”, che i parametri **var** delle procedure sono parametri INOUT realizzati per riferimento, quelli non-**var** sono parametri IN realizzati per copia, con riferimento al programma sottoscritto,

1. per ogni procedura compreso **Esercizio1**, dire quali sono i contenuti del record di attivazione ad essa associato, noti al tempo della compilazione;
2. specificare se tali contenuti differiscono dal caso in cui la politica di assegnazione dell'ambiente non locale sia il “dynamic scoping”;
3. aiutandosi con opportuni diagrammi in cui rappresentare lo stack di attivazione, discutere il comportamento in esecuzione e l'output.

```
program esercizio1 (input, output);
  var a, b, c: integer;

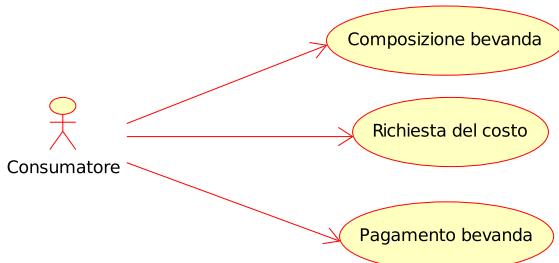
procedure p2(var b: integer);
  var c: integer;
begin
  c:=a+b; b:=b*c; a:=c+b
end;

procedure p1(a: integer);
  procedure p3(var c: integer);
    var a: integer;
  begin
    a:=b+c; b:=b*c; c:=a+b;
    p2(c);
    writeln(a, b, c)
  end;
begin
  b:=a+c; c:=a*c;
  p3(a);
  writeln(a, b, c)
end;

begin
  a:=1; b:=1; c:=2;
  p1(c);
  writeln(a, b, c)
end.
```

## Esercizio 2 – max 15/30

Il distributore automatico fornisce bevande ai consumatori. È dato il diagramma:



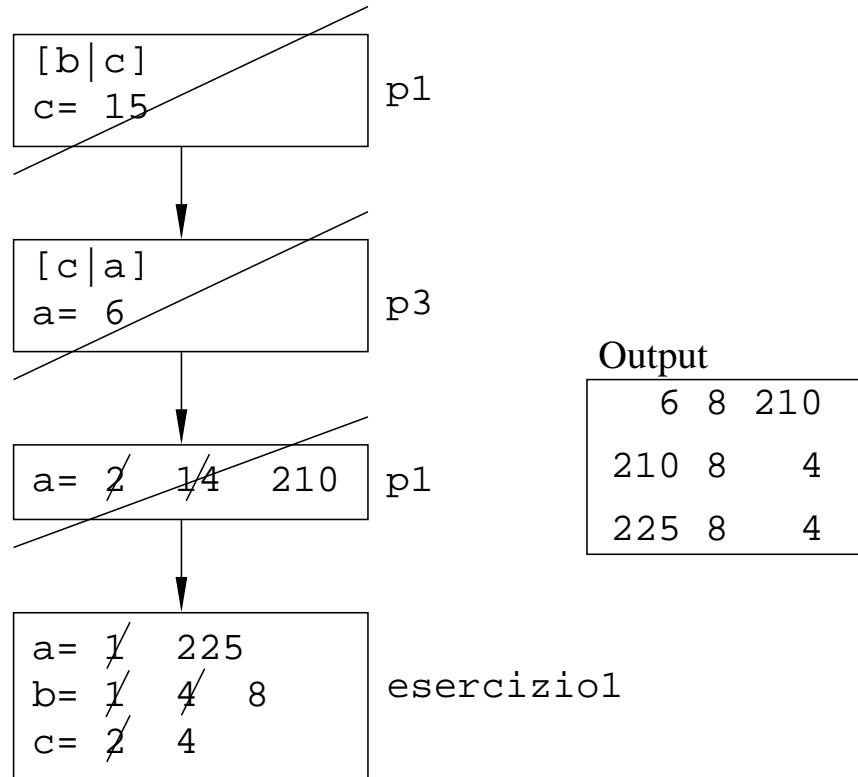
Siamo interessati ad un modello delle bevande. Produrre un diagramma delle classi, tenendo conto che:

- una bevanda ha componenti distinte in componenti principali e componenti opzionali;

- una bevanda ha una sola componente principale corretta eventualmente da più componenti opzionali;
- caffè lungo, espresso, the, orzo, latte, cioccolato, caffè decaffeinato sono componenti principali;
- zucchero, zucchero di canna, miele, cacao, panna, crema di latte sono componenti opzionali;
- ogni componente ha un costo;
- alla fine della scelta delle componenti della bevanda, il consumatore chiede alla bevanda il costo.

Sapendo che il consumatore ha scelto caffè con zucchero, panna e cacao, produrre un diagramma di sequenza per uno scenario del caso d'uso “Richiesta del costo”.

## Soluzione esercizio 1

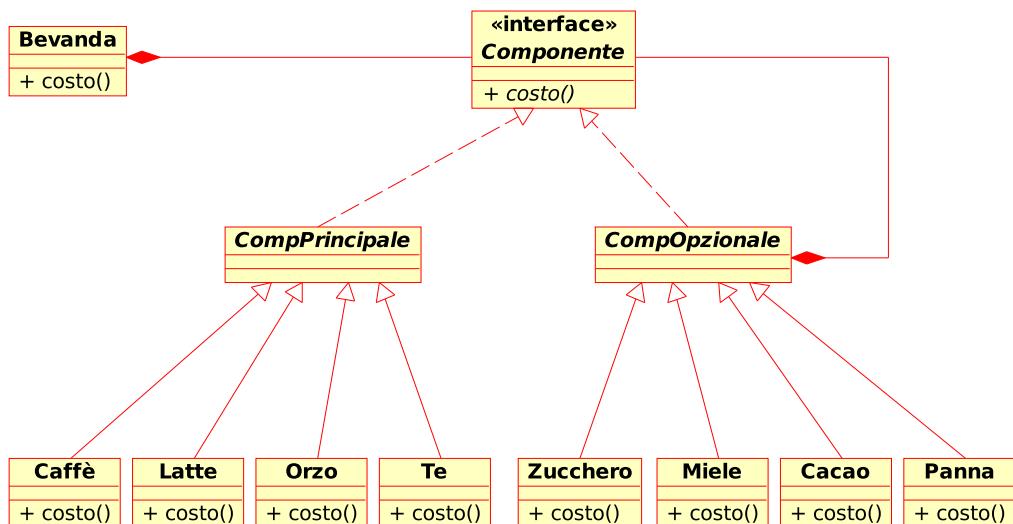


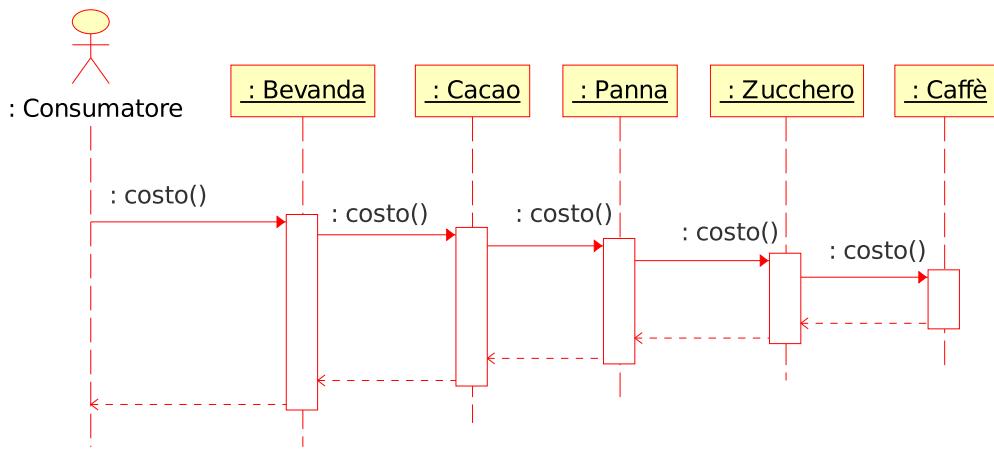
Legenda:

- $[a|b]$ : parametro formale a, parametro attuale b, passaggio INOUT per riferimento.
- $a=4$ : parametro formale a, valore del parametro attuale 4, passaggio IN per copia.

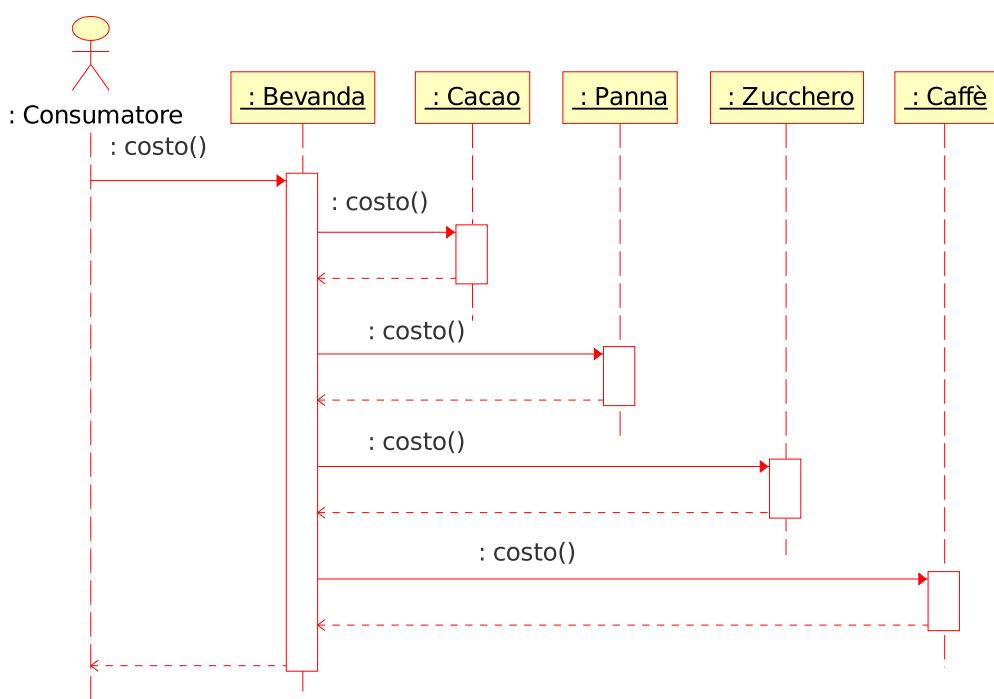
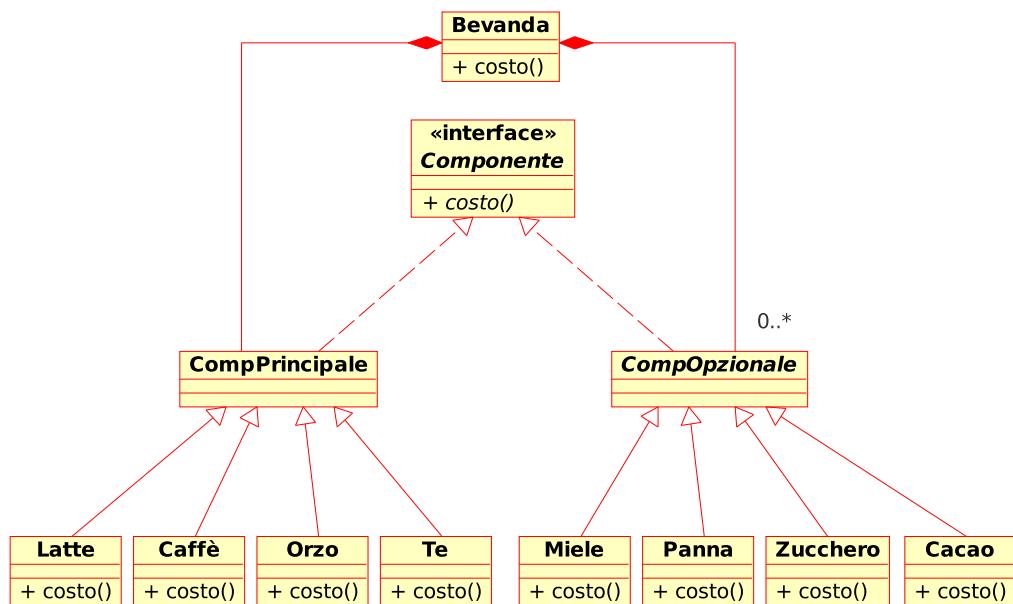
## Soluzione esercizio 2

Possibilità 1

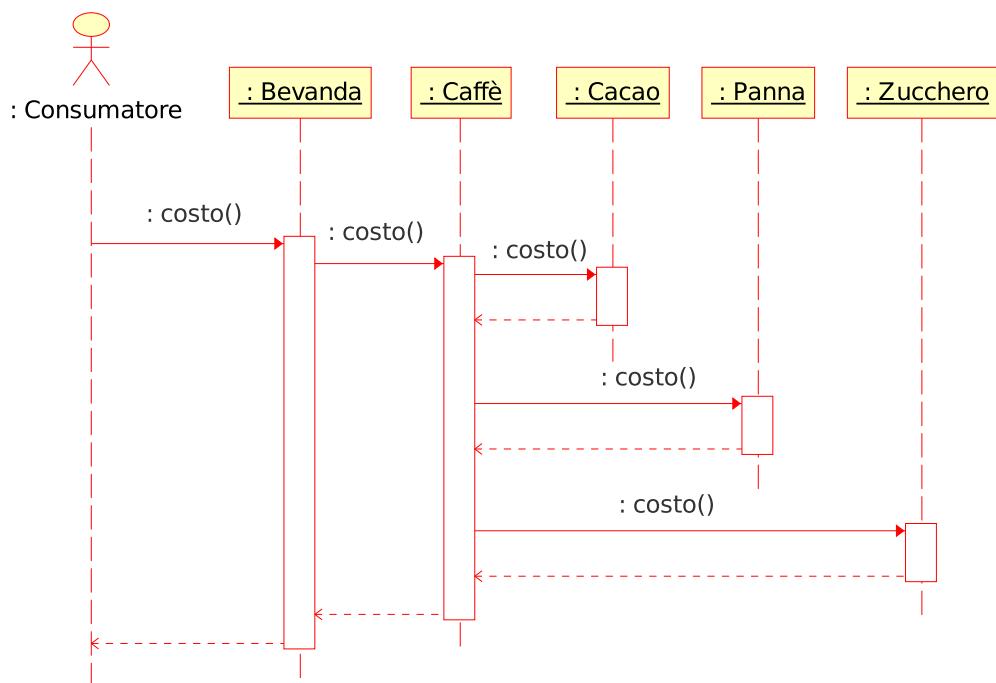
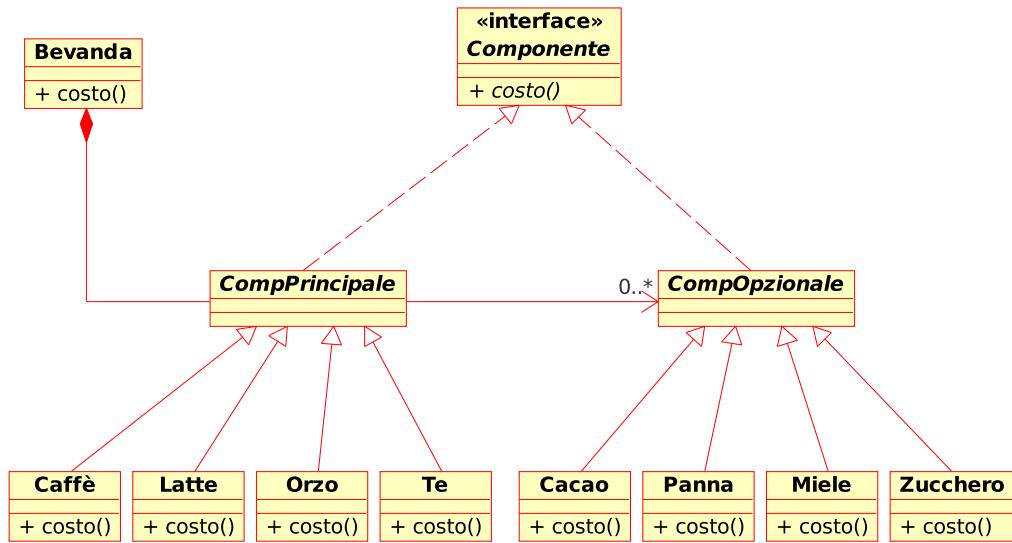




## Possibilità 2



## Possibilità 3



# Prova intercorso di Linguaggi di Programmazione

27 Aprile 2009

*Nome:* \_\_\_\_\_ *Cognome:* \_\_\_\_\_ *Matricola:* \_\_\_\_\_

*Negli esercizi 1 e 2 mettete una X sulle risposte che considerate giuste.*

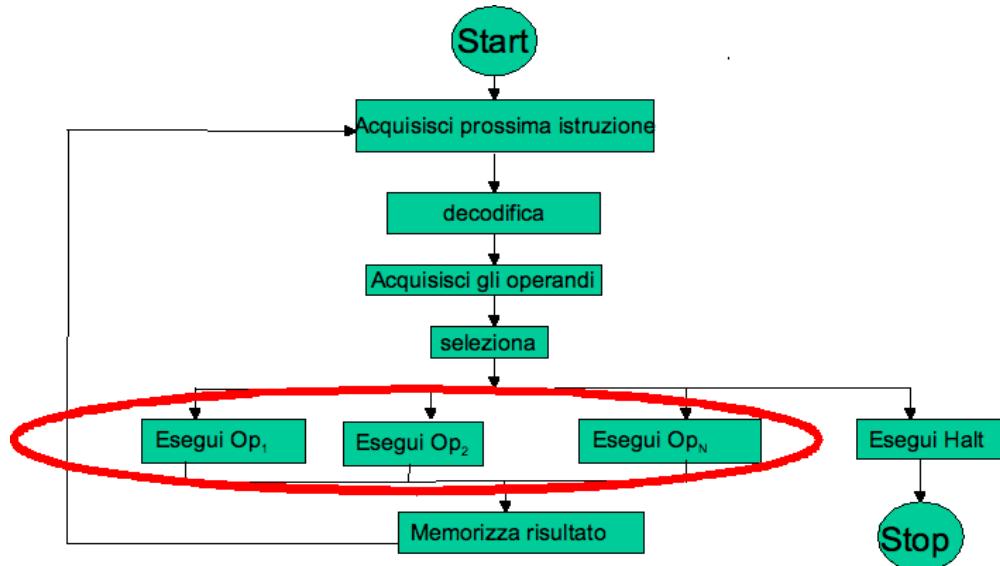
## Esercizio 1

Nella figura seguente, le operazioni che appaiono dentro l'ellisse vanno eseguite tutte in sequenza?



A quale linguaggio di programmazione appartengono?

- (a) Linguaggio di programmazione che state interpretando.  
(b) Linguaggio in cui è scritto l' interprete.  
(c) Linguaggio di programmazione in cui è scritto il programma. (2 punti)



## Esercizio 2

Se un linguaggio è staticamente tipizzato allora:

- (a ) il legame di tipo è definito a compile time ma il controllo di consistenza avviene a run time.
  - (b) il legame di tipo è definito a compile time, il controllo di consistenza avviene sia a run time che a compile time.

Quale fra le due frasi è vera?

## Esercizio 3

**3.1** Fare il diagramma di classe per il seguente problema:

Uno stage teatrale è formato da un copione, degli attori e un regista. Il regista conosce il copione e conosce gli attori.

Un copione è formato da parti, ogni parte è relativa ad un personaggio, in quanto definisce con testi gli interventi verbali del personaggio nell'opera recitata nello stage.

Ogni intervento ha un testo che deve essere recitato dall'attore.

Un attore interpreta una o più parti,

Recitare un copione significa interpretare da parte degli attori le parti rispettive intervento per intervento quando è il momento.

Gli attori sono comici o drammatici.

Gli attori comici quando interpretano una parte producono come effetto, ad ogni intervento, il riso degli spettatori, mentre gli attori drammatici quando interpretano una parte producono, ad ogni intervento, il pianto degli spettatori

(7 punti)

**3.2** Quali diagrammi possono cambiare se nel nostro stage introduciamo altri due attori Francesca e Marta?

(1 punto)

**3.3** Quali diagrammi certamente cambia se introduciamo nel nostro stage un attore che produce l'applauso degli spettatori?

(1 punto)

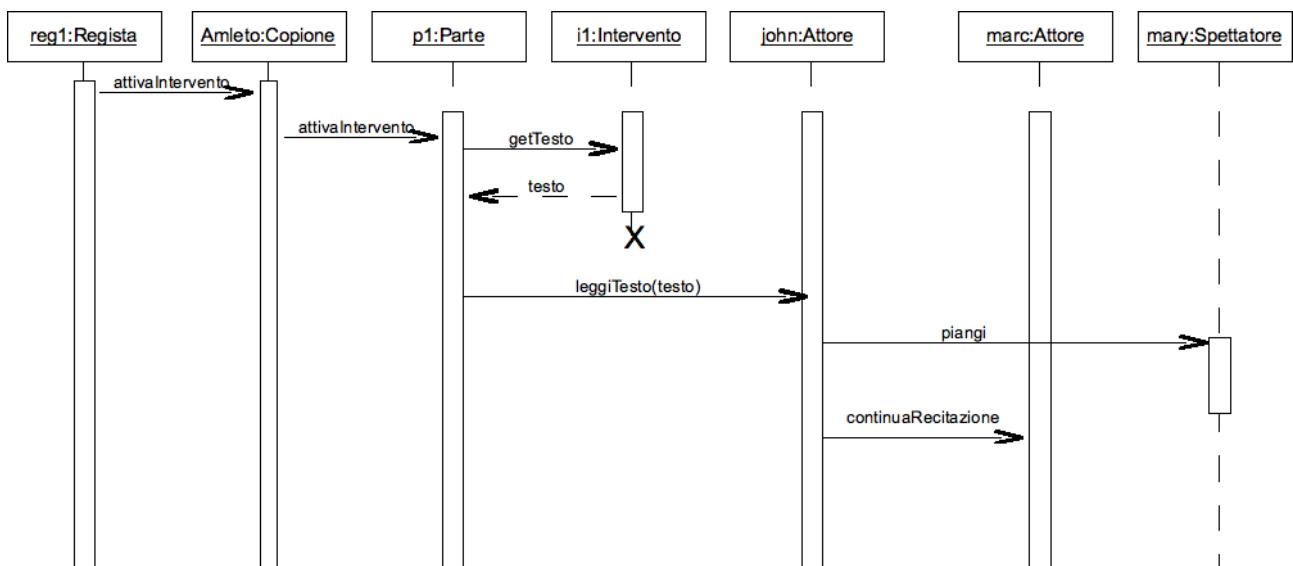
## Esercizio 4

**4.1** Disegnare un sequence diagram per la costruzione di uno Stage caratterizzato da: un attore drammatico John, un attore comico Marc, ed un copione con due parti Parte1 e Parte2 relative a due personaggi: Mario e Carlo, con John che interpreta Parte1 e Marc che interpreta Parte2 con Parte1, relativa a Mario, che contiene 2 interventi, Parte2 relativa a Carlo che contiene 1 intervento.

(4 punti)

**4.2** Dire se il sequence diagram in figura può essere corretto rispetto al diagramma delle classi che avete disegnato nell'esercizio precedente.

(3 punti)



Sequence diagram di attivazione intervento

## Esercizio 5

Supponiamo che la politica di visibilità sia quella dello static scoping (ambito statico) trovare il risultato del seguente programma:

```
program esercizio_imp (input,output)

var x,y,z : integer;

procedure proc1 ( [MODE1] x : integer; [MODE2] y : integer );
begin
    x := x+y;
    if x = z then proc2(y,x) else proc2(x,y)
end;

procedure proc2 ( [MODE3] x, y : integer );
begin
    x := x*y+1;
    if z-y = x then x := 0 else x:= 111
end;

begin
    x :=1; y:= 10; z := 20;
    proc1(z,y);
    writeln(x,y,z)
end;

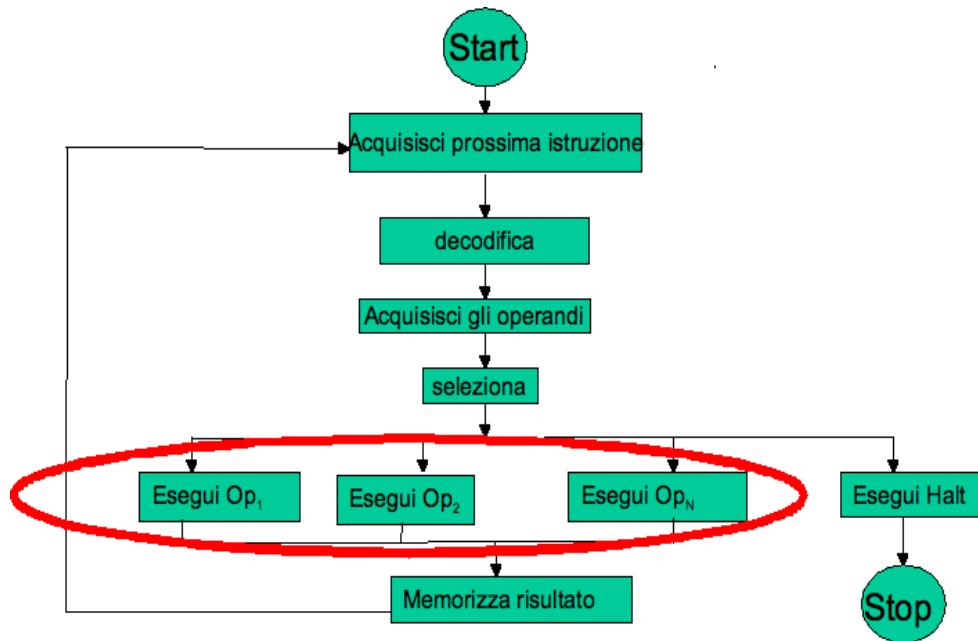
[MODE1] sempre IN per copia
[MODE2] sempre IN OUT per indirizzo
[MODE3] tutti i modi per indirizzo
```

**Tempo assegnato per la soluzione: 2 ore e 30'**

# Soluzioni prova intercorso

27 Aprile 2009

# Esercizio 1

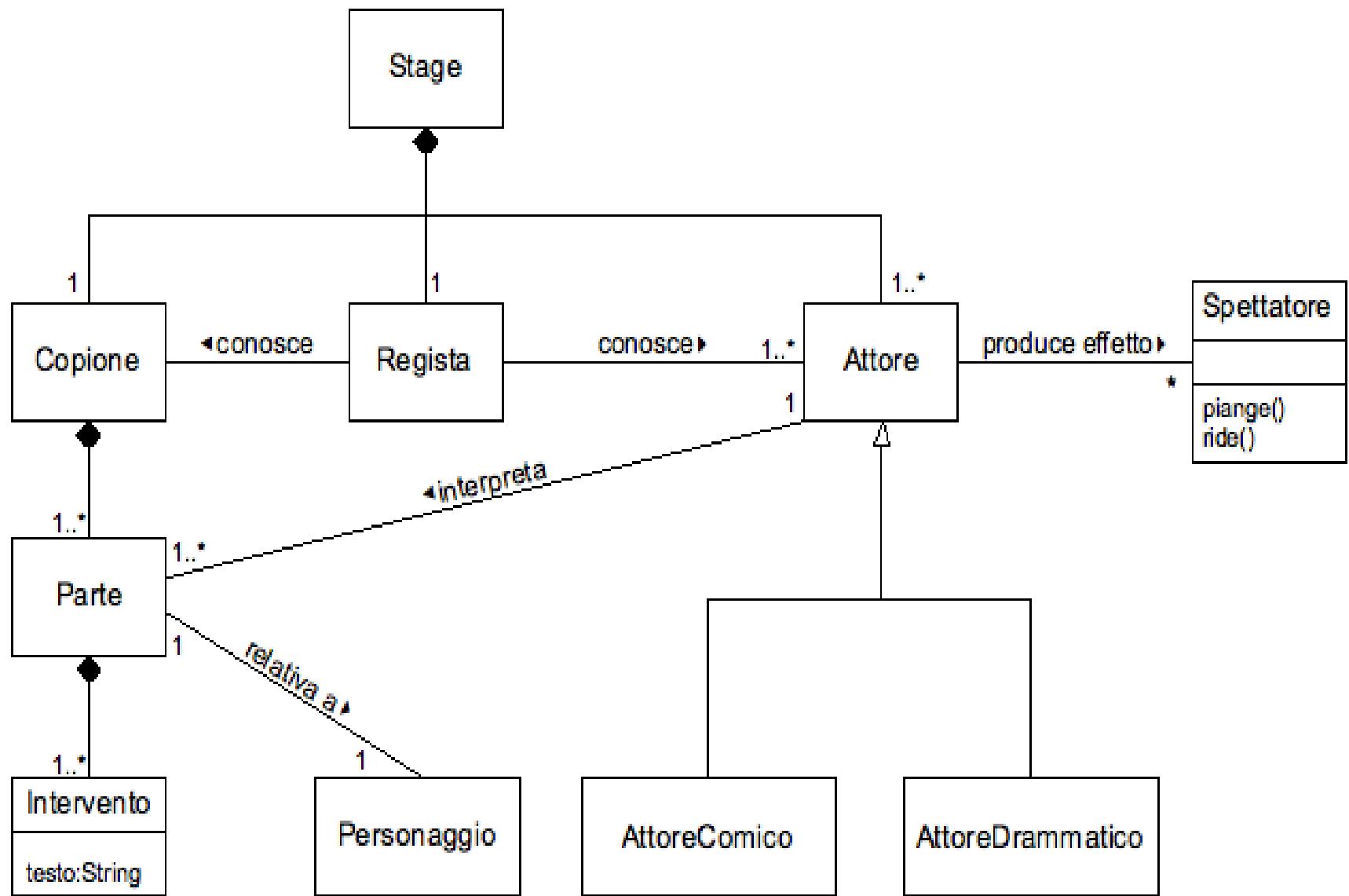


- (a) le operazioni non vengono eseguite in sequenza
- (a) le  $Op_i$  appartengono al linguaggio interpretato

# Esercizio 2

- Se il linguaggio è fortemente tipato
  - il legame di tipo è definito a compile time
  - il controllo di consistenza (in generale) è distribuito tra compile time e run time

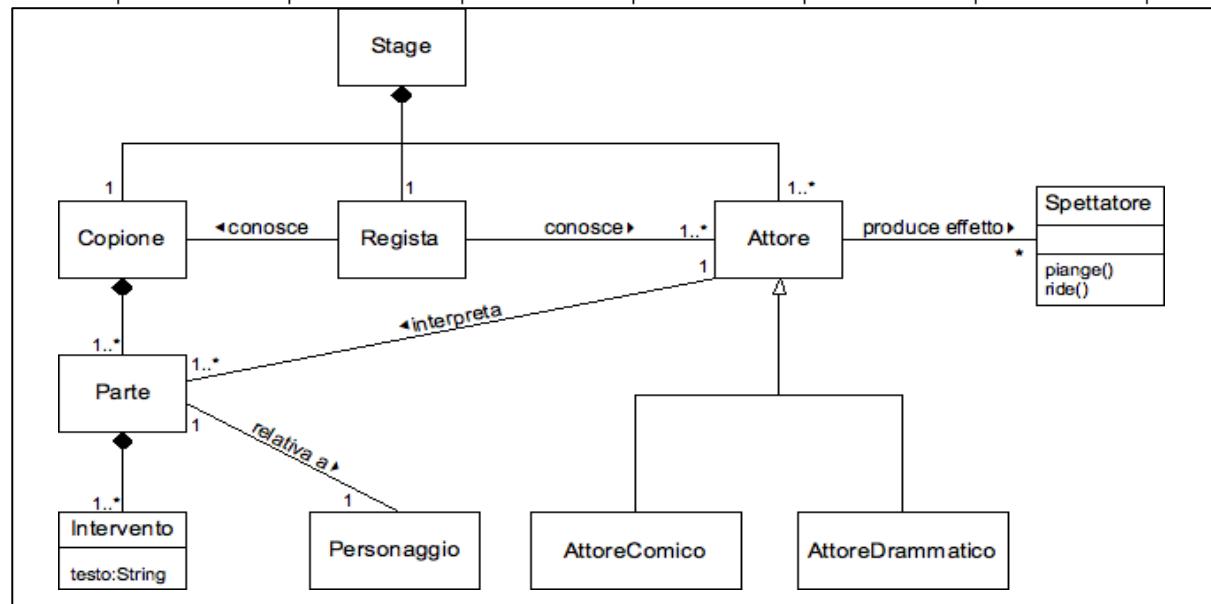
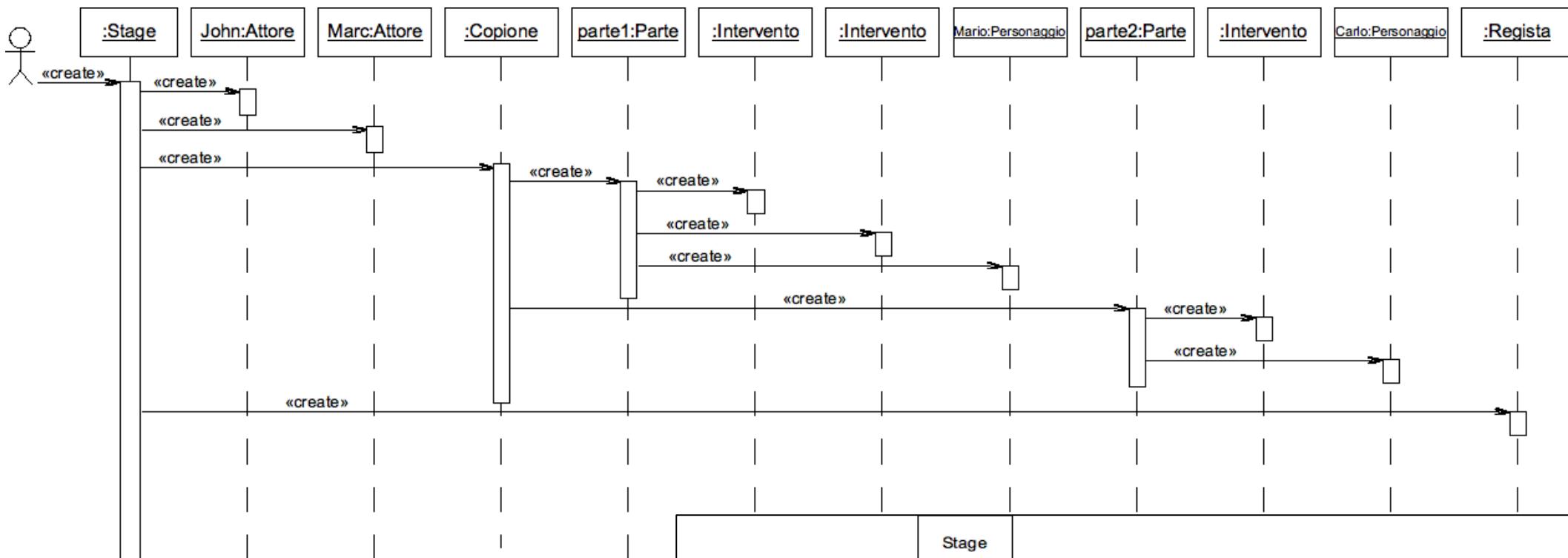
# Esercizio 3.1



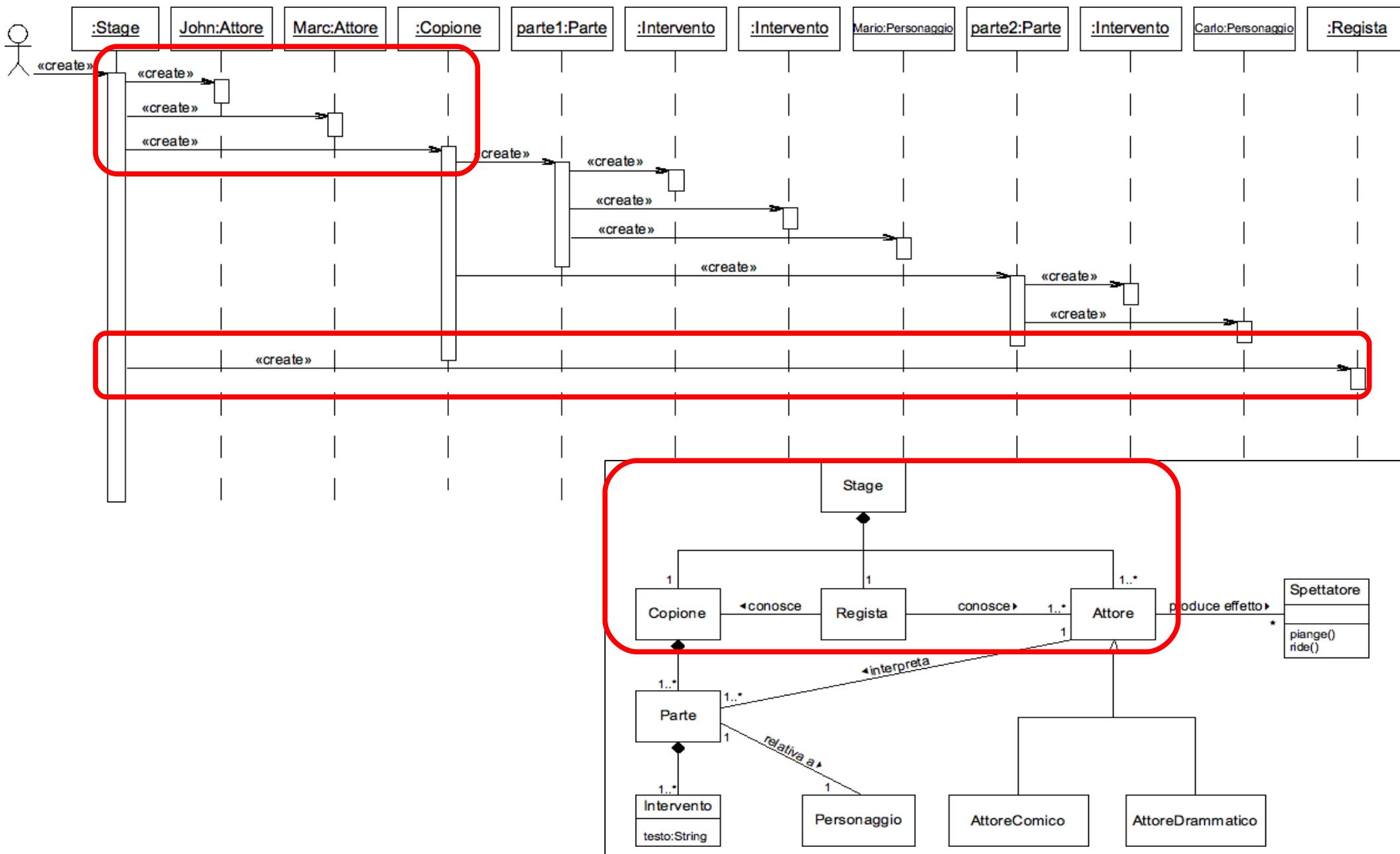
# Esercizio 3 (cont.)

- 3.2 Se introduciamo altri due attori Francesca e Marta:
  - aggiungiamo degli *oggetti*
  - potrebbero quindi cambiare i diagrammi che contengono oggetti:
    - degli oggetti, sequence, collaboration
- 3.3 Se introduciamo attori che possono suscitare applausi
  - aggiungiamo tipi e funzionalità nuovi
  - cambia certamente il diagramma delle classi

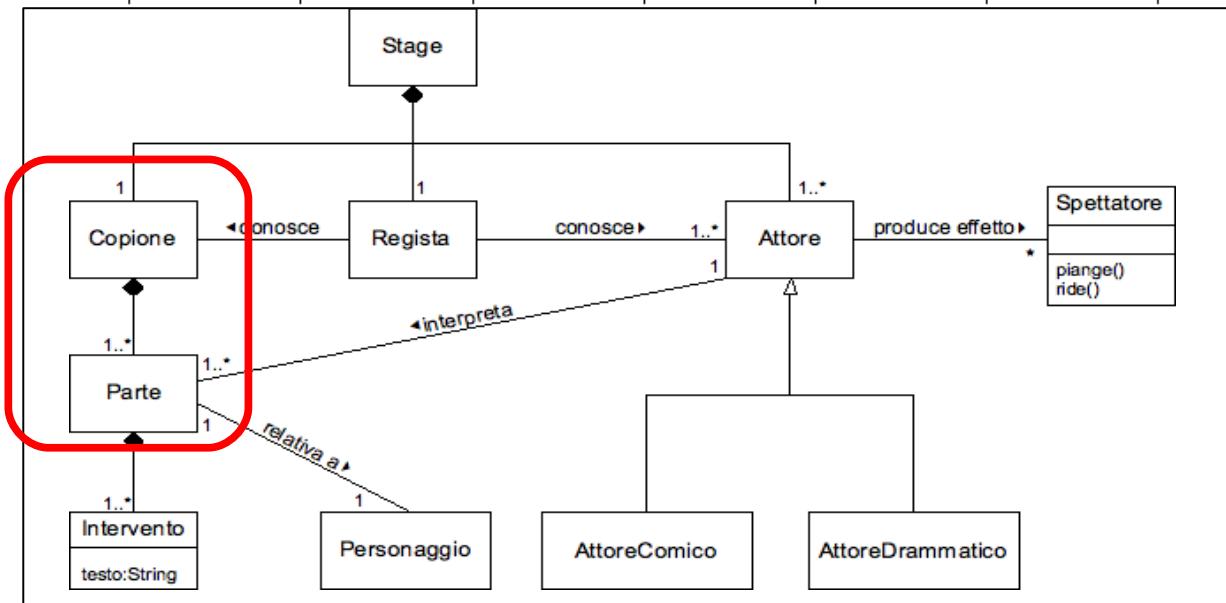
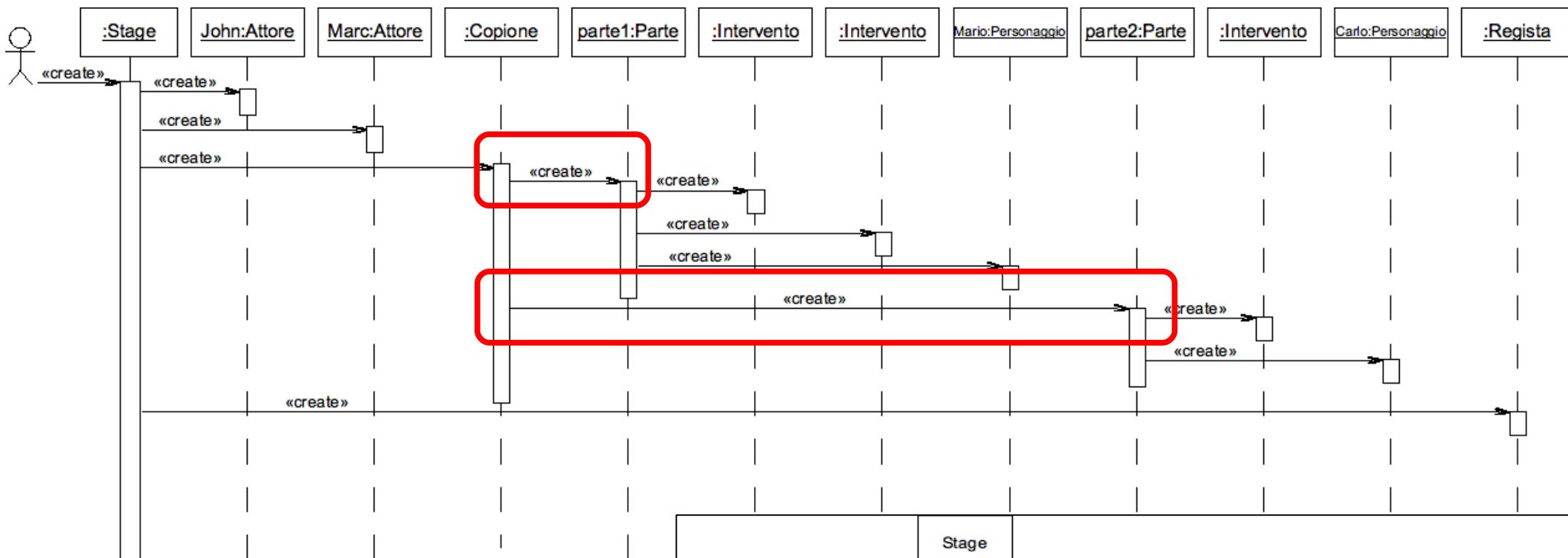
# Esercizio 4.1



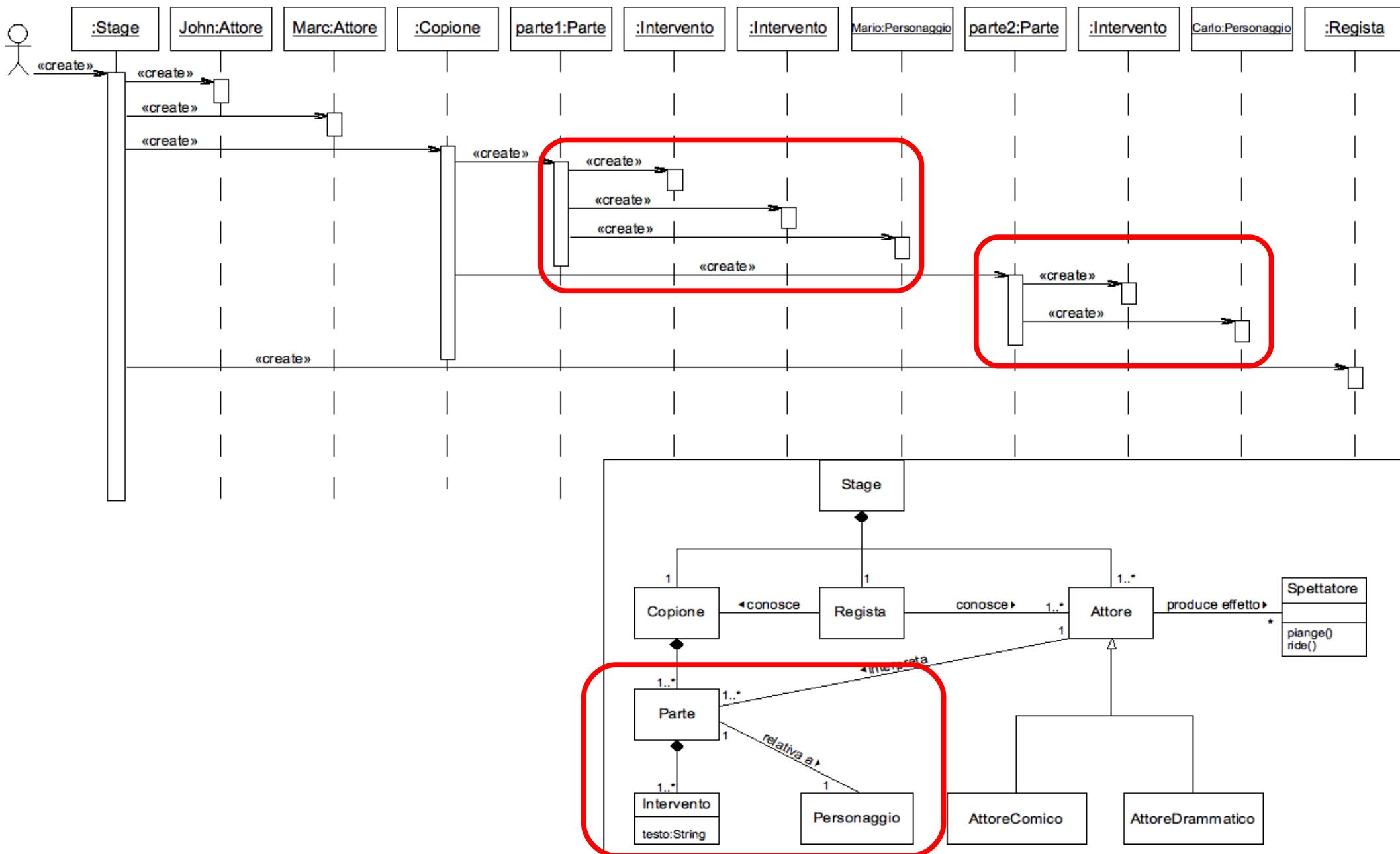
# Esercizio 4.1



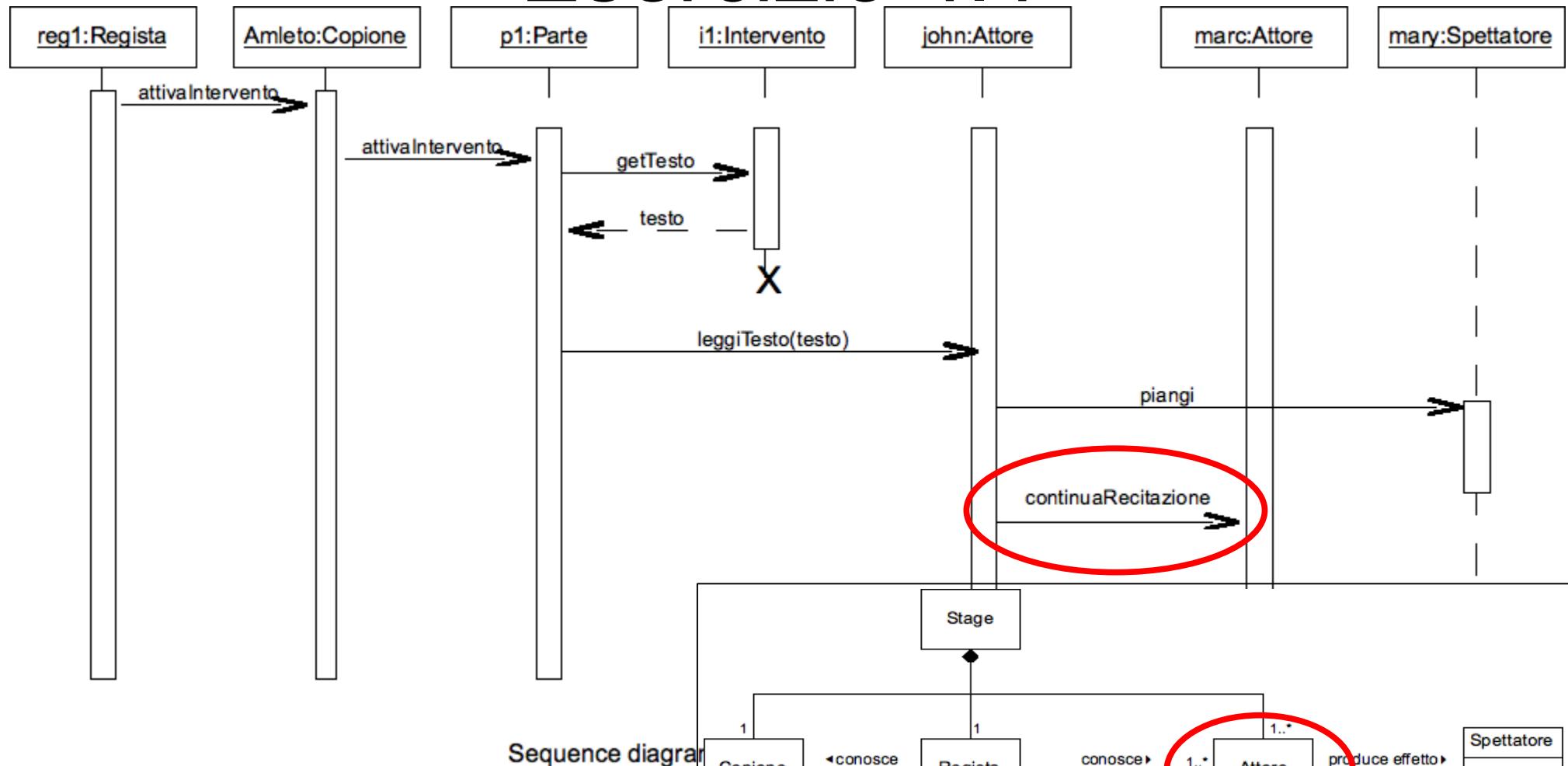
# Esercizio 4.1



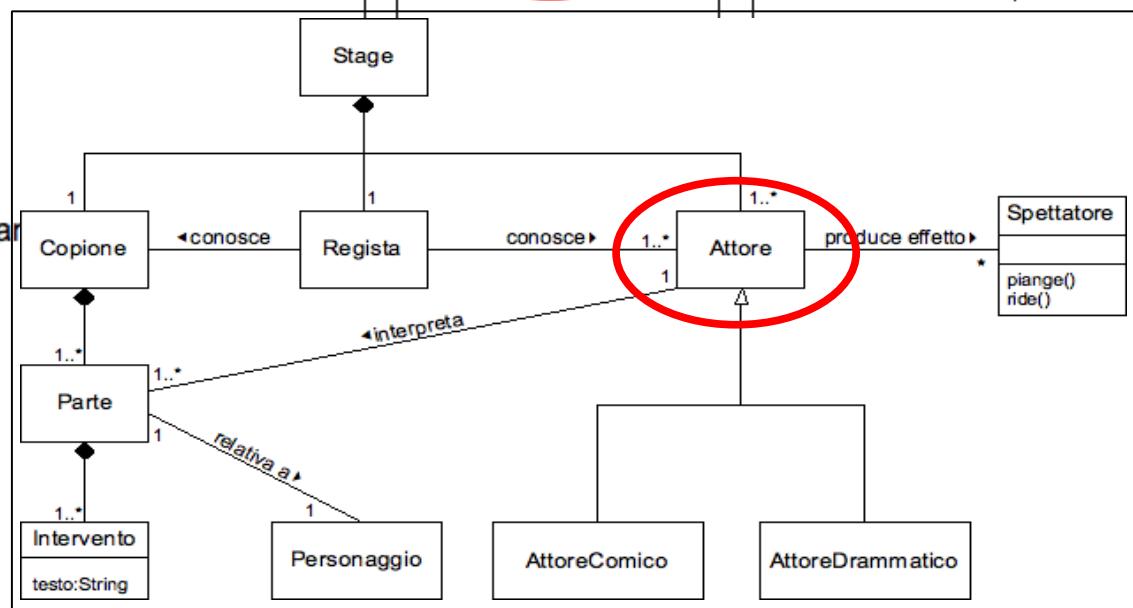
# Esercizio 4.1



# Esercizio 4.1



Sequence diagram



# Esercizio 5

- se in proc2 x è IN per indirizzo
  - ERRORE:  $\textcolor{red}{x} := x * y + 1;$
- se in proc2 x è OUT per indirizzo
  - ERRORE:  $x := \textcolor{red}{x} * y + 1;$
- se in proc2 y è OUT per indirizzo
  - ERRORE:  $x := x * \textcolor{red}{y} + 1;$

# Esercizio 5 (cont.)

- se in proc2 x è IN OUT per indirizzo e y non è OUT: **1 10 20**
  - (nessuna modifica ai valori iniziali)
  - l'unica variabile non locale in proc1 e proc2 è z, che non compare mai a sinistra di un assegnamento nè come parametro OUT o IN OUT, quindi non viene modificata
  - **in proc1**  $x == z+y$  ( $\neq z$ ) quindi si esegue sempre il ramo else ( $proc2(x,y)$ ) che può modificare x (locale a proc1, quindi nessun effetto all'esterno) ma non y ( $proc2$  non modifica il suo secondo parametro)

Dei 16 esercizi dello scritto di Java, 7 rientrano sempre nelle tipologie riportate qui. Segnare nel penultimo foglio le risposte alle seguenti domande a scelta multipla tenendo presente che ogni domanda richiede una sola risposta. I riquadri rossi nell'ultima pagina indicano le risposte corrette.

**Prova n. 1**

1. Date le dichiarazioni:

```
Boolean c;
Error e;
Object n;
```

indicare quali dei seguenti assegnamenti sono corretti a tempo di compilazione.

- A. `c = n;`
- B. `c = (Boolean) e;`
- C. `n = (Object) c;`
- D. `e = n;`
- E. `c = e;`

2. Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class C1 {
    public static void main(String [] argv) throws Exception IOExceptions{C1 {
        try {
            System.out.print(1);
            n();
            System.out.print(2);
        }
        catch( Exception w ) {
            System.out.print(3);
        }
        catch( MyExc2 v ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
        }
    }
    static void n() throws Exception {
        try {
            System.out.print(6);
            throw( new MyExc1() );
        }
        catch( Exception t ) {
            throw( new MyExc2() );
        }
    }
}
```

```
}
catch( MyExc3 h ) {
    System.out.print(7);
}
catch( MyExc1 s ) {
    System.out.print(8);
}
}
```

- A. 16325
- B. Errore a tempo di compilazione
- C. 1635
- D. 1682
- E. Nessuna delle precedenti

3. Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class C1 {
    public static void main(String [] argv) throws Exception IOExceptions{C1 {
        try {
            System.out.print(1);
            m();
            System.out.print(2);
        }
        catch( MyExc3 u ) {
            System.out.print(3);
        }
        catch( MyExc1 i ) {
            System.out.print(4);
            throw( new MyExc2() );
        }
        finally {
            System.out.print(5);
        }
    }
    static void m() throws Exception {
        try {
        }
        catch( MyExc2 f ) {
            System.out.print(6);
        }
    }
}
```

```

        throw( new MyExc2() );
    }
    catch( MyExc1 f ) {
        System.out.print(7);
    }
    finally {
        throw( new MyExc3() );
    }
}

```

- A. 12  
B. Errore a tempo di compilazione  
C. 1325  
D. 135  
E. Nessuna delle precedenti
- 

#### 4. Quale output si ottiene invocando il metodo m?

```

class H {
    private String s1 = "";
    private boolean [] a1 = new boolean [3];
    void m() {
        private Object a3;
        Boolean b2;
        a3 = a1;
        b2 = new Boolean(true);
        p(s1.concat("ab"), a3, b2);
    }
    void p(String s3, Object a2, Boolean b1) {
        String s2;
        Boolean b3;
        s2 = s1.concat("ab");
        b3 = new Boolean(true);
        if(s2 == s3) {
            System.out.print(1);
        } else {
            System.out.print(0);
        }
        if(a1 == a2) {
            System.out.print(1);
        } else {
            System.out.print(0);
        }
        if(b1 == b3) {
            System.out.print(1);
        } else {
            System.out.print(0);
        }
    }
}

```

- A. 101  
B. 110  
C. 001  
D. 011  
E. 010

#### 5. Dire quale delle seguenti affermazioni è falsa:

- A. È possibile dichiarare un array senza indicarne la dimensione
  - B. Un attributo può essere contemporaneamente static e final
  - C. Non è possibile dichiarare un attributo senza inizializzarlo
  - D. Un attributo può essere contemporaneamente static e private
  - E. Un array ha un'unica superclasse
- 

#### 6. Date le dichiarazioni:

```

Object [] m;
Integer [] s;
Boolean [] w;
w = new Boolean [0];
s = new Integer [9];
m = new Integer [7];

```

indicare quali dei seguenti assegnamenti sono corretti a tempo di esecuzione.

- A. w = (Boolean []) m;
  - B. w = (Boolean []) s;
  - C. s = (Integer []) m;
  - D. s = (Integer []) w;
  - E. Nessuno dei precedenti
- 

#### 7. Qual è l'output di questo codice?

```

class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            m();
        }
        catch( MyExc1 h ) {
            throw( new MyExc1() );
        }
        catch( MyExc2 b ) {
            System.out.print(2);
            throw( new MyExc1() );
        }
        finally {
            System.out.print(3);
            throw( new MyExc2() );
        }
    }
    static void m() {
        try {

```

```
System.out.print(4);
throw( new MyExc3() );
}
catch( MyExc1 c ) {
    System.out.print(5);
    throw( new MyExc2() );
}
catch( MyExc3 h ) {
}
catch( MyExc2 u ) {
}
finally {
    System.out.print(6);
}
}
}
throw( new MyExc2() );
}
}
A. 14632... (ciclo infinito)
B. 14
C. 14623Exception in thread main MyExc2
D. Errore a tempo di compilazione
E. Nessuna delle precedenti
```

---

**Prova n. 1**

Università di Napoli Federico II – Corso di Laurea in Informatica

**LP1**

**Prova d'esame**

proff. Piero A. Bonatti e Eliana Minicozzi

5 giugno 2012

Studente e matricola:

Ora di inizio:

Ora di consegna:

|   |   |   |   |   |   |
|---|---|---|---|---|---|
| 1 | A | B | C | D | E |
| 2 | A | B | C | D | E |
| 3 | A | B | C | D | E |
| 4 | A | B | C | D | E |
| 5 | A | B | C | D | E |
| 6 | A | B | C | D | E |
| 7 | A | B | C | D | E |

|   |  |  |   |   |   |   |
|---|--|--|---|---|---|---|
| 1 |  |  |   | ■ |   |   |
| 2 |  |  | ■ |   |   |   |
| 3 |  |  |   |   | ■ |   |
| 4 |  |  |   |   |   | ■ |
| 5 |  |  |   | ■ |   |   |
| 6 |  |  |   | ■ |   |   |
| 7 |  |  |   | ■ |   |   |

Risultato prova n. 1:

Dei 16 esercizi dello scritto di Java, 7 rientrano sempre nelle tipologie riportate qui. Segnare nel penultimo foglio le risposte alle seguenti domande a scelta multipla tenendo presente che ogni domanda richiede una sola risposta. I riquadri rossi nell'ultima pagina indicano le risposte corrette.

## Prova n. 1

1. Date le dichiarazioni:

```
Boolean c;
Error e;
Object n;
```

indicare quali dei seguenti assegnamenti sono corretti a tempo di compilazione.

- A. `c = n;`
- B. `c = (Boolean) e;`
- C. `n = (Object) c;`
- D. `e = n;`
- E. `c = e;`

2. Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class C1 {
    public static void main(String [] argv) throws IOExceptions{C1 {
        try {
            System.out.print(1);
            n();
            System.out.print(2);
        }
        catch( Exception w ) {
            System.out.print(3);
        }
        catch( MyExc2 v ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
        }
    }
    static void n() throws Exception {
        try {
            System.out.print(6);
            throw( new MyExc1() );
        }
        catch( Exception t ) {
            throw( new MyExc2() );
        }
    }
}
```

```
        }
        catch( MyExc3 h ) {
            System.out.print(7);
        }
        catch( MyExc1 s ) {
            System.out.print(8);
        }
    }
}
A. 16325
B. Errore a tempo di compilazione
C. 1635
D. 1682
E. Nessuna delle precedenti
```

3. Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class C1 {
    public static void main(String [] argv) throws IOExceptions{C1 {
        try {
            System.out.print(1);
            m();
            System.out.print(2);
        }
        catch( MyExc3 u ) {
            System.out.print(3);
        }
        catch( MyExc1 i ) {
            System.out.print(4);
            throw( new MyExc2() );
        }
        finally {
            System.out.print(5);
        }
    }
    static void m() throws Exception {
        try {
        }
        catch( MyExc2 f ) {
            System.out.print(6);
        }
    }
}
```

```

        throw( new MyExc2() );
    }
    catch( MyExc1 f ) {
        System.out.print(7);
    }
    finally {
        throw( new MyExc3() );
    }
}

```

- A. 12  
B. Errore a tempo di compilazione  
C. 1325  
D. 135  
E. Nessuna delle precedenti
- 

#### 4. Quale output si ottiene invocando il metodo m?

```

class H {
    private String s1 = "";
    private boolean [] a1 = new boolean [3];
    void m() {
        private Object a3;
        Boolean b2;
        a3 = a1;
        b2 = new Boolean(true);
        p(s1.concat("ab"), a3, b2);
    }
    void p(String s3, Object a2, Boolean b1) {
        String s2;
        Boolean b3;
        s2 = s1.concat("ab");
        b3 = new Boolean(true);
        if(s2 == s3) {
            System.out.print(1);
        } else {
            System.out.print(0);
        }
        if(a1 == a2) {
            System.out.print(1);
        } else {
            System.out.print(0);
        }
        if(b1 == b3) {
            System.out.print(1);
        } else {
            System.out.print(0);
        }
    }
}

```

- A. 101  
B. 110  
C. 001  
D. 011  
E. 010

#### 5. Dire quale delle seguenti affermazioni è falsa:

- A. È possibile dichiarare un array senza indicarne la dimensione
  - B. Un attributo può essere contemporaneamente static e final
  - C. Non è possibile dichiarare un attributo senza inizializzarlo
  - D. Un attributo può essere contemporaneamente static e private
  - E. Un array ha un'unica superclasse
- 

#### 6. Date le dichiarazioni:

```

Object [] m;
Integer [] s;
Boolean [] w;
w = new Boolean [0];
s = new Integer [9];
m = new Integer [7];

```

indicare quali dei seguenti assegnamenti sono corretti a tempo di esecuzione.

- A. w = (Boolean []) m;
  - B. w = (Boolean []) s;
  - C. s = (Integer []) m;
  - D. s = (Integer []) w;
  - E. Nessuno dei precedenti
- 

#### 7. Qual è l'output di questo codice?

```

class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            m();
        }
        catch( MyExc1 h ) {
            throw( new MyExc1() );
        }
        catch( MyExc2 b ) {
            System.out.print(2);
            throw( new MyExc1() );
        }
        finally {
            System.out.print(3);
            throw( new MyExc2() );
        }
    }
    static void m() {
        try {

```

```
System.out.print(4);
throw( new MyExc3() );
}
catch( MyExc1 c ) {
    System.out.print(5);
    throw( new MyExc2() );
}
catch( MyExc3 h ) {
}
catch( MyExc2 u ) {
}
finally {
    System.out.print(6);
}
}
}
throw( new MyExc2() );
}
}
A. 14632... (ciclo infinito)
B. 14
C. 14623Exception in thread main MyExc2
D. Errore a tempo di compilazione
E. Nessuna delle precedenti
```

---

**Prova n. 1**

Università di Napoli Federico II – Corso di Laurea in Informatica

**LP1**

**Prova d'esame**

proff. Piero A. Bonatti e Eliana Minicozzi

5 giugno 2012

Studente e matricola:

Ora di inizio:

Ora di consegna:

|   |   |   |   |   |   |
|---|---|---|---|---|---|
| 1 | A | B | C | D | E |
| 2 | A | B | C | D | E |
| 3 | A | B | C | D | E |
| 4 | A | B | C | D | E |
| 5 | A | B | C | D | E |
| 6 | A | B | C | D | E |
| 7 | A | B | C | D | E |

|   |  |  |   |   |   |   |
|---|--|--|---|---|---|---|
| 1 |  |  |   | ■ |   |   |
| 2 |  |  | ■ |   |   |   |
| 3 |  |  |   |   | ■ |   |
| 4 |  |  |   |   |   | ■ |
| 5 |  |  |   | ■ |   |   |
| 6 |  |  |   | ■ |   |   |
| 7 |  |  |   | ■ |   |   |

Risultato prova n. 1:

Dei 16 esercizi dello scritto di Java, 7 rientrano sempre nelle tipologie riportate qui. Segnare nel penultimo foglio le risposte alle seguenti domande a scelta multipla tenendo presente che ogni domanda richiede una sola risposta. I riquadri rossi nell'ultima pagina indicano le risposte corrette.

## Prova n. 1

1. Qual è l'output di questo codice?

```

class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            q();
        }
        catch( MyExc1 r ) {
        }
        catch( MyExc2 t ) {
            System.out.print(2);
        }
        catch( Exception a ) {
            System.out.print(3);
        }
        finally {
            throw( new MyExc2() );
        }
    }
    static void q() throws Exception {
        try {
            System.out.print(4);
            throw( new Exception() );
        }
        catch( MyExc3 e ) {
        }
        catch( MyExc2 x ) {
            System.out.print(5);
        }
        finally {
            System.out.print(6);
        }
    }
}

```

- A. 146
- B. 1463
- C. Errore a tempo di compilazione
- D. 1463Exception in thread main MyExc2
- E. Nessuna delle precedenti

2. Dire quale delle seguenti affermazioni è falsa:

- A. Prima dell'esecuzione del costruttore alle variabili di istanza sono assegnati i valori di default
- B. È possibile dichiarare un attributo senza inizializzarlo
- C. La dimensione di un array può non essere indicata al momento della dichiarazione dell'array
- D. Non è possibile dichiarare un array senza indicarne la dimensione
- E. La lunghezza di un array non può essere variata dopo la sua creazione

3. Quale output si ottiene invocando il metodo q?

```

class A {
    private String s1 = "";
    private String s3;
    private String s4 = new String("abcd");
    void q() {
        Boolean b3 = new Boolean(true);
        private String s6 = s4;
        p(b3, b3, s1.concat(""), s6);
    }
    void p(Boolean b1, Boolean b2, String s2,
           String s5) {
        s3 = s1.concat("");
        if(b2 == b1) {
            System.out.print(1);
        } else {
            System.out.print(0);
        }
        if(s2 == s3) {
            System.out.print(1);
        } else {
            System.out.print(0);
        }
        if(s5 == s4) {
            System.out.print(1);
        } else {

```

```

        System.out.print(0);
    }
}

```

- A. 010
  - B. 101
  - C. 100
  - D. 110
  - E. 001
- 

**4. Qual è l'output di questo codice?**

```

class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class B1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            m();
        }
        catch( MyExc2 g ) {
        }
        catch( Exception g ) {
            System.out.print(2);
        }
        finally {
            System.out.print(3);
            throw( new Exception() );
        }
    }
    static void m() throws Exception {
        try {
            System.out.print(4);
            throw( new MyExc2() );
        }
        catch( Exception v ) {
            System.out.print(5);
        }
        finally {
            System.out.print(6);
        }
    }
}

```

- A. 14563Exception in thread main java.lang.Exception
  - B. 146
  - C. Errore a tempo di compilazione
  - D. 14563
  - E. Nessuna delle precedenti
- 

```

class B2 extends A {...}
class A extends Object {...}
class C1 extends A {...}

```

e le inizializzazioni di variabile:

```

C1 d;
A u;
B2 w;
d = new C1();
w = new B2();
u = new B2();

```

indicare quali dei seguenti assegnamenti sono corretti a tempo di esecuzione.

- A. d = (C1) u;
  - B. d = (C1) w;
  - C. w = (B2) d;
  - D. w = (B2) u;
  - E. Nessuno dei precedenti
- 

**6. Qual è l'output di questo codice?**

```

class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv) {
        try {
            m();
        }
        catch( MyExc3 k ) {
            System.out.print(1);
        }
        catch( Exception u ) {
        }
        finally {
            throw( new Exception() );
        }
    }
    static void m() {
        try {
            System.out.print(2);
            throw( new MyExc3() );
        }
        catch( MyExc3 z ) {
            System.out.print(3);
        }
        catch( Exception d ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
        }
    }
}

```

**5. Date le dichiarazioni:**

- A. 235Exception in thread main java.lang.Exception

- B. 235
  - C. 23
  - D. Errore a tempo di compilazione
  - E. Nessuna delle precedenti
- 

7. Date le dichiarazioni:

```
Boolean n;  
Error q;  
Object y;
```

---

indicare quali dei seguenti assegnamenti sono corretti a tempo di compilazione.

- A. n = q;
  - B. n = (Boolean) q;
  - C. n = y;
  - D. y = (Object) n;
  - E. q = y;
-

**Prova n. 1**

Università di Napoli Federico II – Corso di Laurea in Informatica

**LP1**

**Prova d'esame**

proff. Piero A. Bonatti e Eliana Minicozzi

5 giugno 2012

Studente e matricola:

Ora di inizio:

Ora di consegna:

|   |   |   |   |   |   |
|---|---|---|---|---|---|
| 1 | A | B | C | D | E |
| 2 | A | B | C | D | E |
| 3 | A | B | C | D | E |
| 4 | A | B | C | D | E |
| 5 | A | B | C | D | E |
| 6 | A | B | C | D | E |
| 7 | A | B | C | D | E |

|   |  |  |  |  |  |  |
|---|--|--|--|--|--|--|
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |

Risultato prova n. 1:

### **Versione 1 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Una classe non interna può essere dichiarata `private`
- B) Non tutti i tipi di eccezioni estendono la classe `Throwable`
- C) Quando due metodi hanno lo stesso nome non sempre si ha overloading
- D) Una classe non può essere dichiarata `protected`
- E) In Java non tutti i tipi numerici sono con segno

## **Versione 2 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Un attributo può essere contemporaneamente **static** e **private**
- B) Un attributo **static** può essere acceduto mediante un riferimento a un oggetto della sua classe di appartenenza
- C) Subito dopo l'esecuzione del costruttore alle variabili di istanza sono assegnati i valori di default
- D) Due metodi possono avere diverso nome ma lo stesso tipo di parametri
- E) La dimensione di un array può non essere indicata al momento della dichiarazione dell'array

### **Versione 3 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Un attributo **static** non può essere acceduto mediante il nome della sua classe di appartenenza
- B) Due metodi non possono avere diverso nome ma lo stesso tipo di parametri
- C) È possibile dichiarare un attributo senza inizializzarlo
- D) Ogni valore in memoria non è associato ad un tipo di dato particolare
- E) I modificatori applicati a una variabile di tipo array non si applicano alla variabile array ma ai suoi elementi

#### **Versione 4 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Un oggetto esiste sempre dopo la sua dichiarazione
- B) I modificatori applicati a una variabile di tipo array non si applicano alla variabile array ma ai suoi elementi
- C) Due metodi non possono differire per il tipo di ritorno
- D) In Java non tutti i tipi numerici sono con segno
- E) Nell'overloading due metodi possono avere lo stesso nome ma diverso tipo di parametri

### **Versione 5 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Un array ha un'unica superclasse
- B) Nell'overloading due metodi possono avere lo stesso nome ma diverso tipo di parametri
- C) È possibile dichiarare un attributo senza inizializzarlo
- D) Un oggetto può non esistere dopo la sua dichiarazione
- E) La lunghezza di un array può essere variata dopo la sua creazione

### **Versione 6 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) La lunghezza di un array può essere variata dopo la sua creazione
- B) Un array ha un'unica superclasse
- C) Nell'overloading due metodi possono avere lo stesso nome e lo stesso numero di parametri
- D) Una classe può essere dichiarata `final`
- E) Ai metodi `static` non si applica il *dynamic method dispatch*

### **Versione 7 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Non tutti i tipi di eccezioni estendono la classe `RuntimeException`
- B) La lunghezza di un array non può essere variata dopo la sua creazione
- C) Una classe può essere dichiarata `final`
- D) Un array con riferimento `null` ha lunghezza zero
- E) I modificatori applicati a una variabile di tipo array si applicano alla variabile array e non ai suoi elementi

### **Versione 8 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Un array vuoto può avere riferimento `null`
- B) La dichiarazione di un oggetto e la sua creazione possono essere svolte solo in tempi diversi
- C) Un array ha più di una superclasse
- D) Un attributo non può essere contemporaneamente `static` e `final`
- E) Due metodi possono differire per il tipo di ritorno

### **Versione 9 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Non tutti i tipi di eccezioni estendono la classe `RuntimeException`
- B) Un array con riferimento `null` ha lunghezza zero
- C) Nell'overloading due metodi possono avere lo stesso nome ma diverso tipo di parametri
- D) Nell'overloading due metodi possono avere lo stesso nome e lo stesso numero di parametri
- E) Un attributo può essere contemporaneamente `static` e `final`

### **Versione 10 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Due metodi possono differire per il tipo di ritorno
- B) Un array ha più di una superclasse
- C) La lunghezza di un array può essere variata dopo la sua costruzione
- D) Il minimo numero di elementi che un array può contenere è 1
- E) Ai metodi `static` si applica il *dynamic method dispatch*

### **Versione 1 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Il minimo numero di elementi che un array può contenere è 0
- B) La lunghezza di un array non può essere variata dopo la sua creazione
- C) I modificatori applicati a una variabile di tipo array si applicano alla variabile array e non ai suoi elementi
- D) Ai metodi **static** si applica il *dynamic method dispatch*
- E) Un array con riferimento **null** non ha lunghezza

## **Versione 2 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Un array ha un'unica superclasse
- B) In Java tutti i tipi numerici sono con segno
- C) L'operatore `new` restituisce un riferimento all'oggetto appena creato
- D) I modificatori applicati a una variabile di tipo array non si applicano alla variabile array ma ai suoi elementi
- E) Due metodi possono differire per il tipo di ritorno

### **Versione 3 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Una classe può essere dichiarata `protected`
- B) Ogni valore in memoria è associato ad un tipo di dato particolare
- C) Non tutti i tipi di eccezioni estendono la classe `Object`
- D) Un attributo `static` può essere acceduto mediante un riferimento a un oggetto della sua classe di appartenenza
- E) Un array ha un'unica superclasse

#### **Versione 4 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) La dichiarazione di un oggetto e la sua creazione possono essere svolte solo in tempi diversi
- B) Nell'overloading due metodi non possono avere lo stesso nome ma diverso tipo di parametri
- C) Una classe può essere dichiarata **protected**
- D) Un array ha più di una superclasse
- E) In Java non tutti i tipi numerici sono con segno

### **Versione 5 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Una classe può essere dichiarata **private**
- B) La lunghezza di un array non può essere variata dopo la sua costruzione
- C) La lunghezza di un array può essere variata dopo la sua creazione
- D) Nell'overloading due metodi possono avere lo stesso nome ma diverso tipo di parametri
- E) La dimensione di un array può non essere indicata al momento della dichiarazione dell'array

### **Versione 6 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Non tutti i tipi di eccezioni estendono la classe `RuntimeException`
- B) Un attributo `static` può essere acceduto mediante il nome della sua classe di appartenenza
- C) Non è possibile dichiarare un array senza indicarne la dimensione
- D) Due metodi possono avere diverso nome ma lo stesso tipo di parametri
- E) Ogni valore in memoria è associato ad un tipo di dato particolare

### **Versione 7 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Nell'overloading due metodi non possono avere lo stesso nome ma diverso tipo di parametri
- B) Non è possibile dichiarare un attributo senza inizializzarlo
- C) Quando due metodi hanno lo stesso nome non sempre si ha overloading
- D) Una classe non può essere dichiarata **protected**
- E) La lunghezza di un array può essere variata dopo la sua costruzione

### **Versione 8 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Un oggetto può non esistere dopo la sua dichiarazione
- B) Quando due metodi hanno lo stesso nome non sempre si ha overloading
- C) Nell'overloading due metodi possono avere lo stesso nome e lo stesso numero di parametri
- D) Un attributo **static** non può essere acceduto mediante un riferimento a un oggetto della sua classe di appartenenza
- E) Un array vuoto non può avere riferimento **null**

### **Versione 9 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Una classe non può essere dichiarata `final`
- B) Un array vuoto non può avere riferimento `null`
- C) La lunghezza di un array può essere variata dopo la sua creazione
- D) La dichiarazione di un oggetto e la sua creazione possono essere svolte solo in tempi diversi
- E) Una classe non può essere dichiarata `abstract` o `final`

### **Versione 10 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Non tutti i tipi di eccezioni estendono la classe `Throwable`
- B) Ai metodi `static` non si applica il *dynamic method dispatch*
- C) Una classe non può essere dichiarata `private`
- D) La lunghezza di un array può essere variata dopo la sua creazione
- E) Un array ha più di una superclasse

### **Versione 11 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) La dimensione di un array può non essere indicata al momento della dichiarazione dell'array
- B) I modificatori applicati a una variabile di tipo array non si applicano alla variabile array ma ai suoi elementi
- C) Un attributo può essere contemporaneamente `static` e `final`
- D) Una classe può essere dichiarata `private`
- E) Tutti i tipi di eccezioni estendono la classe `Object`

### **Versione 1 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Un attributo **static** non può essere acceduto mediante un riferimento a un oggetto della sua classe di appartenenza
- B) Una classe non può essere dichiarata **final**
- C) Un attributo può essere contemporaneamente **static** e **final**
- D) Un attributo non può essere contemporaneamente **static** e **private**
- E) In Java non tutti i tipi numerici sono con segno

## **Versione 2 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Non tutti i tipi di eccezioni estendono la classe `RuntimeException`
- B) Una classe non può essere dichiarata `abstract` o `final`
- C) Ogni valore in memoria è associato ad un tipo di dato particolare
- D) Un attributo `static` può essere acceduto mediante un riferimento a un oggetto della sua classe di appartenenza
- E) È possibile dichiarare un attributo senza inizializzarlo

### **Versione 3 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Una classe non interna può essere dichiarata `private`
- B) Due metodi non possono avere diverso nome ma lo stesso tipo di parametri
- C) I modificatori applicati a una variabile di tipo array si applicano alla variabile array e non ai suoi elementi
- D) La dichiarazione di un oggetto e la sua creazione possono essere svolte solo in tempi diversi
- E) Un attributo non può essere contemporaneamente `static` e `private`

#### **Versione 4 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Un oggetto può non esistere dopo la sua dichiarazione
- B) Non è possibile dichiarare un array senza indicarne la dimensione
- C) La dichiarazione di un oggetto e la sua creazione possono essere svolte solo contemporaneamente
- D) Il minimo numero di elementi che un array può contenere è 1
- E) Quando due metodi hanno lo stesso nome si ha overloading

### **Versione 5 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) La dichiarazione di un oggetto e la sua creazione possono essere svolte solo contemporaneamente
- B) Un attributo **static** non può essere acceduto mediante un riferimento a un oggetto della sua classe di appartenenza
- C) Un array ha più di una superclasse
- D) Un array ha un'unica superclasse
- E) Una classe non interna può essere dichiarata **private**

### **Versione 6 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) La dimensione di un array deve essere indicata al momento della dichiarazione dell'array
- B) La lunghezza di un array può essere variata dopo la sua creazione
- C) Due metodi non possono avere diverso nome ma lo stesso tipo di parametri
- D) Non tutti i tipi di eccezioni estendono la classe `RuntimeException`
- E) Un attributo `static` non può essere acceduto mediante il nome della sua classe di appartenenza

### **Versione 7 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) In Java tutti i tipi numerici sono con segno
- B) Un array con riferimento `null` non ha lunghezza
- C) Tutti i tipi di eccezioni estendono la classe `Throwable`
- D) È possibile dichiarare un attributo senza inizializzarlo
- E) Subito dopo l'esecuzione del costruttore alle variabili di istanza sono assegnati i valori di default

### **Versione 8 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Due metodi non possono avere diverso nome ma lo stesso tipo di parametri
- B) Ai metodi `static` non si applica il *dynamic method dispatch*
- C) Non è possibile dichiarare un attributo senza inizializzarlo
- D) Tutti i tipi di eccezioni estendono la classe `RuntimeException`
- E) Un array ha più di una superclasse

### **Versione 9 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Non tutti i tipi di eccezioni estendono la classe `Throwable`
- B) Tutti i tipi di eccezioni estendono la classe `Object`
- C) Un array vuoto non può avere riferimento `null`
- D) La dichiarazione di un oggetto e la sua creazione possono essere svolte in tempi diversi
- E) Un array con riferimento `null` non ha lunghezza

### **Versione 10 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) L'operatore `new` restituisce un riferimento all'oggetto appena creato
- B) Un array possiede dei membri
- C) Una classe non può essere dichiarata `final`
- D) Il minimo numero di elementi che un array può contenere è 0
- E) Ogni valore in memoria è associato ad un tipo di dato particolare

### **Versione 11 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Un oggetto può non esistere dopo la sua dichiarazione
- B) Un array ha più di una superclasse
- C) Un array vuoto non può avere riferimento `null`
- D) Un array ha un'unica superclasse
- E) La lunghezza di un array non può essere variata dopo la sua creazione

### **Versione 12 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) In Java tutti i tipi numerici sono con segno
- B) Subito dopo l'esecuzione del costruttore alle variabili di istanza sono assegnati i valori di default
- C) Ai metodi **static** si applica il *dynamic method dispatch*
- D) Una classe non interna può essere dichiarata **private**
- E) La dichiarazione di un oggetto e la sua creazione possono essere svolte solo contemporaneamente

### **Versione 13 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) La lunghezza di un array può essere variata dopo la sua creazione
- B) Nell'overloading due metodi non possono avere lo stesso nome ma diverso tipo di parametri
- C) Un oggetto esiste sempre dopo la sua dichiarazione
- D) Un array ha più di una superclasse
- E) Ogni valore in memoria è associato ad un tipo di dato particolare

### **Versione 14 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Una classe può essere dichiarata `abstract` o `final`
- B) L'operatore `new` restituisce un riferimento all'oggetto appena creato
- C) Un array con riferimento `null` non ha lunghezza
- D) Una classe non interna non può essere dichiarata `private`
- E) Subito dopo l'esecuzione del costruttore alle variabili di istanza sono assegnati i valori di default

### **Versione 15 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Un attributo non può essere contemporaneamente **static** e **final**
- B) L'operatore **new** restituisce un riferimento all'oggetto appena creato
- C) Non tutti i tipi di eccezioni estendono la classe **Object**
- D) Un array non possiede dei membri
- E) Una classe non può essere dichiarata **abstract** o **final**

### **Versione 16 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Una classe può essere dichiarata **protected**
- B) In Java non tutti i tipi numerici sono con segno
- C) Nell'overloading due metodi non possono avere lo stesso nome ma diverso tipo di parametri
- D) Non è possibile dichiarare un attributo senza inizializzarlo
- E) Subito dopo l'esecuzione del costruttore alle variabili di istanza sono assegnati i valori di default

### **Versione 17 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Un array ha più di una superclasse
- B) Nell'overloading due metodi non possono avere lo stesso nome e lo stesso numero di parametri
- C) Quando due metodi hanno lo stesso nome non sempre si ha overloading
- D) Un array vuoto può avere riferimento `null`
- E) Un array non possiede dei membri

### **Versione 18 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Quando due metodi hanno lo stesso nome si ha overloading
- B) Tutti i tipi di eccezioni estendono la classe `Throwable`
- C) Due metodi possono differire per il tipo di ritorno
- D) La lunghezza di un array non può essere variata dopo la sua creazione
- E) Prima dell'esecuzione del costruttore alle variabili di istanza sono assegnati i valori di default

### **Versione 1 dell'esercizio 2**

Date le dichiarazioni:

```
class Super extends Object {...}
class B1 extends Super {...}
class B extends Super {...}
```

e le inizializzazioni di variabile:

```
B c;
Super g;
B1 m;
g = new B();
c = new B();
m = new B1();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) `c = (B) g;`
- B) `m = (B1) g;`
- C) `c = (B) m;`
- D) `m = (B1) c;`
- E) Nessuno dei precedenti

## Versione 2 dell'esercizio 2

Date le dichiarazioni:

```
class A extends Object {...}
class B2 extends A {...}
class C1 extends A {...}
```

e le inizializzazioni di variabile:

```
C1 c;
B2 d;
A n;
n = new C1();
d = new B2();
c = new C1();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) `c = (C1) n;`
- B) `d = (B2) n;`
- C) `c = (C1) d;`
- D) `d = (B2) c;`
- E) Nessuno dei precedenti

### **Versione 3 dell'esercizio 2**

Date le dichiarazioni:

```
class A extends Object {...}
class C1 extends A {...}
class B1 extends A {...}
```

e le inizializzazioni di variabile:

```
C1 c;
A e;
B1 q;
c = new C1();
e = new C1();
q = new B1();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) q = (B1) c;
- B) c = (C1) e;
- C) q = (B1) e;
- D) c = (C1) q;
- E) Nessuno dei precedenti

#### **Versione 4 dell'esercizio 2**

Date le dichiarazioni:

```
class B2 extends C0 {...}
class C1 extends C0 {...}
class C0 extends Object {...}
```

e le inizializzazioni di variabile:

```
C1 m;
C0 t;
B2 v;
m = new C1();
t = new C1();
v = new B2();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) `v = (B2) t;`
- B) `v = (B2) m;`
- C) `m = (C1) t;`
- D) `m = (C1) v;`
- E) Nessuno dei precedenti

### **Versione 5 dell'esercizio 2**

Date le dichiarazioni:

```
class B2 extends Super {...}  
class C2 extends Super {...}  
class Super extends Object {...}
```

e le inizializzazioni di variabile:

```
B2 c;  
Super h;  
C2 v;  
v = new C2();  
c = new B2();  
h = new C2();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) v = (C2) c;
- B) c = (B2) v;
- C) v = (C2) h;
- D) c = (B2) h;
- E) Nessuno dei precedenti

### **Versione 6 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] [] d;  
Object [] f;  
Integer [] s;  
f = new Object [8] [6];  
s = new Integer [5];  
d = new Object [8] [1];
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) `d = (Object []) [] s;`
- B) `d = (Object []) [] f;`
- C) `s = (Integer []) f;`
- D) `s = (Integer []) d;`
- E) Nessuno dei precedenti

### **Versione 7 dell'esercizio 2**

Date le dichiarazioni:

```
class C1 extends Super {...}  
class Super extends Object {...}  
class C2 extends Super {...}
```

e le inizializzazioni di variabile:

```
Super m;  
C2 n;  
C1 s;  
m = new C2();  
s = new C1();  
n = new C2();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) s = (C1) m;
- B) n = (C2) m;
- C) n = (C2) s;
- D) s = (C1) n;
- E) Nessuno dei precedenti

### **Versione 8 dell'esercizio 2**

Date le dichiarazioni:

```
Error c;  
Exception g;  
Object s;  
g = new Exception();  
s = new Exception();  
c = new Error();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) c = (Error) s;
- B) g = (Exception) s;
- C) g = (Exception) c;
- D) c = (Error) g;
- E) Nessuno dei precedenti

### **Versione 9 dell'esercizio 2**

Date le dichiarazioni:

```
Boolean n;  
Object u;  
String v;  
v = new String("");  
n = new Boolean(true);  
u = new String("abcdef");
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) v = (String) n;
- B) n = (Boolean) u;
- C) n = (Boolean) v;
- D) v = (String) u;
- E) Nessuno dei precedenti

## Versione 10 dell'esercizio 2

Date le dichiarazioni:

```
class C0 extends Object {...}
class B extends C0 {...}
class C2 extends C0 {...}
```

e le inizializzazioni di variabile:

```
B m;
C2 q;
C0 w;
q = new C2();
m = new B();
w = new C2();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) m = (B) w;
- B) q = (C2) m;
- C) m = (B) q;
- D) q = (C2) w;
- E) Nessuno dei precedenti

### **Versione 1 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] g;  
Error [] m;  
Exception [] u;  
u = new Exception [4];  
g = new Error [6];  
m = new Error [2];
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) m = (Error []) u;
- B) m = (Error []) g;
- C) u = (Exception []) g;
- D) u = (Exception []) m;
- E) Nessuno dei precedenti

## **Versione 2 dell'esercizio 2**

Date le dichiarazioni:

```
Object m;  
Boolean q;  
Integer z;  
z = new Integer(10);  
m = new Integer(1);  
q = new Boolean(true);
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) q = (Boolean) z;
- B) z = (Integer) q;
- C) z = (Integer) m;
- D) q = (Boolean) m;
- E) Nessuno dei precedenti

### **Versione 3 dell'esercizio 2**

Date le dichiarazioni:

```
String [] c;
Integer [] x;
Object [] z;
c = new String [1];
z = new Integer [4];
x = new Integer [3];
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) `x = (Integer []) z;`
- B) `c = (String []) z;`
- C) `c = (String []) x;`
- D) `x = (Integer []) c;`
- E) Nessuno dei precedenti

#### **Versione 4 dell'esercizio 2**

Date le dichiarazioni:

```
Object c;
Exception m;
String w;
w = new String("abcd");
c = new String("abcd");
m = new Exception();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) m = (Exception) c;
- B) w = (String) c;
- C) w = (String) m;
- D) m = (Exception) w;
- E) Nessuno dei precedenti

### **Versione 5 dell'esercizio 2**

Date le dichiarazioni:

```
String [] b;  
Object [] g;  
Exception [] u;  
g = new String [2];  
b = new String [4];  
u = new Exception [9];
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) `b = (String []) u;`
- B) `b = (String []) g;`
- C) `u = (Exception []) b;`
- D) `u = (Exception []) g;`
- E) Nessuno dei precedenti

## Versione 6 dell'esercizio 2

Date le dichiarazioni:

```
class B1 extends A {...}
class C1 extends A {...}
class A extends Object {...}
```

e le inizializzazioni di variabile:

```
C1 b;
A g;
B1 q;
b = new C1();
g = new B1();
q = new B1();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) q = (B1) b;
- B) b = (C1) g;
- C) q = (B1) g;
- D) b = (C1) q;
- E) Nessuno dei precedenti

### **Versione 7 dell'esercizio 2**

Date le dichiarazioni:

```
class C2 extends A {...}  
class A extends Object {...}  
class Sub2 extends A {...}
```

e le inizializzazioni di variabile:

```
Sub2 c;  
A e;  
C2 n;  
n = new C2();  
c = new Sub2();  
e = new C2();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) n = (C2) e;
- B) c = (Sub2) n;
- C) c = (Sub2) e;
- D) n = (C2) c;
- E) Nessuno dei precedenti

### **Versione 8 dell'esercizio 2**

Date le dichiarazioni:

```
Object c;
Boolean f;
Error m;
c = new Boolean(false);
m = new Error();
f = new Boolean(true);
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) m = (Error) c;
- B) f = (Boolean) m;
- C) f = (Boolean) c;
- D) m = (Error) f;
- E) Nessuno dei precedenti

### **Versione 9 dell'esercizio 2**

Date le dichiarazioni:

```
class Sub1 extends A {...}  
class A extends Object {...}  
class C2 extends A {...}
```

e le inizializzazioni di variabile:

```
Sub1 h;  
A m;  
C2 t;  
h = new Sub1();  
m = new Sub1();  
t = new C2();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) t = (C2) h;
- B) h = (Sub1) t;
- C) t = (C2) m;
- D) h = (Sub1) m;
- E) Nessuno dei precedenti

## Versione 10 dell'esercizio 2

Date le dichiarazioni:

```
class Sub1 extends C0 {...}
class C0 extends Object {...}
class B2 extends C0 {...}
```

e le inizializzazioni di variabile:

```
B2 r;
C0 s;
Sub1 z;
z = new Sub1();
r = new B2();
s = new B2();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) `r = (B2) s;`
- B) `z = (Sub1) r;`
- C) `r = (B2) z;`
- D) `z = (Sub1) s;`
- E) Nessuno dei precedenti

### **Versione 11 dell'esercizio 2**

Date le dichiarazioni:

```
Integer [] u;  
String [] w;  
Object [] x;  
w = new String [9];  
x = new String [0];  
u = new Integer [8];
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) w = (String []) x;
- B) w = (String []) u;
- C) u = (Integer []) x;
- D) u = (Integer []) w;
- E) Nessuno dei precedenti

### **Versione 1 dell'esercizio 2**

Date le dichiarazioni:

```
class A extends Object {...}  
class B extends A {...}  
class C2 extends A {...}
```

e le dichiarazioni di variabile:

```
C2 m;  
B y;  
A z;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `m = (C2) y;`
- B) `z = (A) y;`
- C) `m = z;`
- D) `y = z;`
- E) `y = (B) m;`

## **Versione 2 dell'esercizio 2**

Date le dichiarazioni:

```
Exception [] g;  
String [] h;  
Object [] s;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `g = h;`
- B) `h = g;`
- C) `h = (String []) g;`
- D) `g = (Exception []) h;`
- E) `h = (String []) s;`

### **Versione 3 dell'esercizio 2**

Date le dichiarazioni:

```
Integer h;  
Object s;  
Exception z;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `z = h;`
- B) `z = (Exception) h;`
- C) `h = (Integer) s;`
- D) `z = s;`
- E) `h = z;`

#### **Versione 4 dell'esercizio 2**

Date le dichiarazioni:

```
Object c;  
String p;  
Integer w;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `w = c;`
- B) `p = w;`
- C) `w = (Integer) p;`
- D) `c = p;`
- E) `p = (String) w;`

### **Versione 5 dell'esercizio 2**

Date le dichiarazioni:

```
Integer b;  
String e;  
Object h;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `b = (Integer) e;`
- B) `h = b;`
- C) `e = (String) b;`
- D) `e = h;`
- E) `b = e;`

### **Versione 6 dell'esercizio 2**

Date le dichiarazioni:

```
Boolean [] p;  
Object [] [] r;  
Object [] s;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `p = r;`
- B) `r = s;`
- C) `r = (Object []) [] p;`
- D) `r = p;`
- E) `s = p;`

### **Versione 7 dell'esercizio 2**

Date le dichiarazioni:

```
Object b;  
Exception n;  
Boolean t;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) n = b;
- B) n = t;
- C) t = (Boolean) n;
- D) t = (Boolean) b;
- E) n = (Exception) t;

### **Versione 8 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] b;  
Object [] [] d;  
Boolean [] y;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `y = (Boolean []) d;`
- B) `b = d;`
- C) `d = y;`
- D) `y = b;`
- E) `d = (Object [] []) y;`

### **Versione 9 dell'esercizio 2**

Date le dichiarazioni:

```
class Super extends Object {...}  
class C2 extends Super {...}  
class Sub1 extends Super {...}
```

e le dichiarazioni di variabile:

```
C2 b;  
Sub1 g;  
Super n;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `b = g;`
- B) `g = n;`
- C) `b = n;`
- D) `n = g;`
- E) `g = (Sub1) b;`

### **Versione 10 dell'esercizio 2**

Date le dichiarazioni:

```
Boolean f;  
Integer g;  
Object m;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `f = m;`
- B) `g = f;`
- C) `g = (Integer) f;`
- D) `f = (Boolean) g;`
- E) `g = (Integer) m;`

### **Versione 11 dell'esercizio 2**

Date le dichiarazioni:

```
Object a;  
Exception m;  
Boolean n;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) m = n;
- B) m = (Exception) a;
- C) m = a;
- D) n = m;
- E) n = a;

### **Versione 12 dell'esercizio 2**

Date le dichiarazioni:

```
Boolean a;  
Object q;  
String y;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) a = y;
- B) a = q;
- C) a = (Boolean) q;
- D) y = q;
- E) a = (Boolean) y;

### **Versione 13 dell'esercizio 2**

Date le dichiarazioni:

```
Integer g;  
Error s;  
Object u;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `g = (Integer) s;`
- B) `s = u;`
- C) `s = (Error) u;`
- D) `g = s;`
- E) `s = g;`

### **Versione 14 dell'esercizio 2**

Date le dichiarazioni:

```
Boolean a;  
String r;  
Object x;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) a = (Boolean) r;
- B) r = a;
- C) x = (Object) a;
- D) a = r;
- E) a = x;

### **Versione 15 dell'esercizio 2**

Date le dichiarazioni:

```
Object h;  
String m;  
Exception u;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) u = m;
- B) m = (String) u;
- C) u = (Exception) m;
- D) h = (Object) u;
- E) m = h;

### **Versione 16 dell'esercizio 2**

Date le dichiarazioni:

```
class C0 extends Object {...}
class B2 extends C0 {...}
class Sub1 extends C0 {...}
```

e le dichiarazioni di variabile:

```
Sub1 e;
B2 v;
C0 x;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) v = (B2) e;
- B) e = x;
- C) v = x;
- D) v = e;
- E) x = (C0) e;

### **Versione 17 dell'esercizio 2**

Date le dichiarazioni:

```
String f;  
Object q;  
Boolean z;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `z = f;`
- B) `q = (Object) z;`
- C) `f = (String) z;`
- D) `f = z;`
- E) `z = q;`

### **Versione 18 dell'esercizio 2**

Date le dichiarazioni:

```
class C0 extends Object {...}
class Sub1 extends C0 {...}
class C2 extends C0 {...}
```

e le dichiarazioni di variabile:

```
C0 e;
Sub1 f;
C2 r;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `r = (C2) f;`
- B) `f = e;`
- C) `f = r;`
- D) `r = (C2) e;`
- E) `r = e;`

### **Versione 1 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In una implementazione interpretativa pura l'interprete produce un file con il codice oggetto a partire dal sorgente?

- A) si
- B) no

ii) Il polimorfismo per inclusione è meno indicato di quello parametrico per la definizione di collezioni di oggetti omogenei?

- A) si
- B) no

### **Versione 2 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In un programma che usa record varianti (come le union del C) il controllo dei tipi può essere fatto interamente a tempo di compilazione?

- A) si
- B) no

ii) C adotta lo scoping statico?

- A) si
- B) no

### **Versione 3 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Il primo garbage collector è stato introdotto

- A) in Java
- B) in C#
- C) in Lisp

ii) Nel primo Fortran esistevano già dei costrutti per allocare memoria a runtime?

- A) si
- B) no

### **Versione 4 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) I parametri IN passati per riferimento
  - A) non possono essere letti prima di essere inizializzati
  - B) non possono essere modificati
  - C) nessuna delle precedenti
  
- ii) Il Lisp ha un garbage collector?
  - A) si
  - B) no

### **Versione 5 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) La proprietà caratterizzante del paradigma a oggetti è:
  - A) l'encapsulation
  - B) la gerarchia di tipi
  - C) le interfacce
  
- ii) Una lista di oggetti i cui tipi non sono confrontabili tra loro si può definire con il polimorfismo parametrico (templates)?
  - A) si
  - B) no

### **Versione 6 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) SQL è un linguaggio di programmazione general purpose?

- A) si
- B) no

ii) Java ha una implementazione:

- A) compilata
- B) interpretata
- C) mista

### **Versione 7 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Il C++ ha una implementazione:
  - A) compilata
  - B) interpretata
  - C) mista
  
- ii) Se implementiamo le liste mediante template, i membri di una lista hanno tutti lo stesso tipo (o nel caso object oriented delle sottoclassi di quel tipo)?
  - A) si
  - B) no

### **Versione 8 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Il C++ ha un garbage collector?
  - A) si
  - B) no
- ii) Quando viene invocata una macro (come le #define in C), viene inserito un record di attivazione nello stack?
  - A) si
  - B) no
  - C) dipende

### **Versione 9 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Tra i compiti del supporto a run time c'è la gestione della memoria?
  - A) si
  - B) no
  
- ii) Il predicato Prolog member può essere utilizzato sia per verificare se un dato elemento appartiene a una lista sia come "generatore", cioè per enumerare tutti i membri della lista uno a uno?
  - A) si
  - B) no

### **Versione 10 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Nel primo Fortran esistevano già dei costrutti per allocare memoria a runtime?

- A) si
- B) no

ii) È vero che nel paradigma funzionale puro non ci sono gli assegnamenti?

- A) si
- B) no

### **Versione 1 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Se in Java non si usa il polimorfismo per inclusione, allora tutti i controlli di tipo possono avvenire a tempo di compilazione?

- A) si
- B) no

ii) C adotta la structural equivalence per le struct e la name equivalence per tutti gli altri tipi?

- A) si
- B) no

### **Versione 2 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Il controllo di correttezza dei downcast richiede controlli a runtime?

- A) si
- B) no
- C) dipende

ii) C adotta la structural equivalence per le struct e la name equivalence per tutti gli altri tipi?

- A) si
- B) no

### **Versione 3 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Un tipo di dato astratto è
  - A) una classe astratta
  - B) una interfaccia
  - C) un tipo perfettamente incapsulato
  
- ii) Il primo garbage collector è stato introdotto
  - A) in Java
  - B) in C#
  - C) in Lisp

### **Versione 4 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Il C supporta l'equivalenza strutturale
  - A) sui tipi primitivi
  - B) sulle struct
  
- ii) Il C adotta sempre la name equivalence tra tipi?
  - A) si
  - B) no

### **Versione 5 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) In Java l'ambiente non locale di una classe interna a un'altra classe si può trovare
  - A) sullo stack
  - B) nello heap
  - C) nella zona statica
  
- ii) L'ambiente associa gli identificatori direttamente al loro valore nei linguaggi:
  - A) imperativi
  - B) funzionali
  - C) fortemente tipizzati

### **Versione 6 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Nei linguaggi con implementazione ibrida compilata/interpretata l'input del programma viene elaborato
  - A) dal compilatore
  - B) dall'interprete
- ii) In C le variabili non locali di funzioni e procedure si possono trovare:
  - A) sullo stack
  - B) nello heap
  - C) nella zona statica

### **Versione 7 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In un linguaggio che supporta il polimorfismo per inclusione il controllo dei tipi può essere interamente statico?

- A) si
- B) no

ii) In un linguaggio debolmente tipato il controllo dei tipi avviene sempre interamente a tempo di compilazione

- A) si
- B) no

### **Versione 8 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Tra i compiti del supporto a run time c'è la gestione della memoria?
  - A) si
  - B) no
  
- ii) Un linguaggio senza alcuna forma di cicli (for, while, do-while) può essere computazionalmente completo?
  - A) si
  - B) no

### **Versione 9 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Esistono linguaggi dove le modalità di passaggio dei parametri non vengono fissate al momento della compilazione

- A) si
- B) no

ii) Il C adotta sempre la structural equivalence tra tipi?

- A) si
- B) no

### **Versione 10 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) I parametri IN OUT passati per riferimento
  - A) non possono essere letti prima di essere inizializzati
  - B) non possono essere modificati
  - C) nessuna delle precedenti
  
- ii) In un linguaggio staticamente tipato il controllo dei tipi avviene sempre interamente a tempo di compilazione
  - A) si
  - B) no

### **Versione 11 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Java adotta lo scoping statico?
  - A) si
  - B) no
  
- ii) C adotta la name equivalence per le struct e la structural equivalence per tutti gli altri tipi?
  - A) si
  - B) no

### **Versione 1 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Nei linguaggi con implementazione ibrida compilata/interpretata l'input del programma viene elaborato
  - A) dal compilatore
  - B) dall'interprete
  
- ii) È vero che nel paradigma funzionale puro non ci sono gli assegnamenti?
  - A) si
  - B) no

### **Versione 2 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In Java le variabili non locali dei metodi si possono trovare:

- A) sullo stack
- B) nello heap
- C) nella zona statica

ii) Nei linguaggi funzionali,  $\text{env}(x)$  restituisce

- A) una locazione
- B) il valore di  $x$

### **Versione 3 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) In Java l'ambiente non locale di una classe interna a un metodo si può trovare
  - A) sullo stack
  - B) nello heap
  - C) nella zona statica
  
- ii) I parametri IN passati per copia
  - A) non possono essere letti prima di essere inizializzati
  - B) non possono essere modificati
  - C) nessuna delle precedenti

### **Versione 4 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Il C adotta sempre la name equivalence tra tipi?
  - A) si
  - B) no
  
- ii) La proprietà caratterizzante del paradigma a oggetti è:
  - A) l'encapsulation
  - B) la gerarchia di tipi
  - C) le interfacce

### **Versione 5 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Il predicato Prolog member può essere utilizzato sia per verificare se un dato elemento appartiene a una lista sia come “generatore”, cioè per enumerare tutti i membri della lista uno a uno?
  - A) si
  - B) no
- ii) Un tipo di dato astratto è
  - A) una classe astratta
  - B) una interfaccia
  - C) un tipo perfettamente encapsulato

### **Versione 6 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In un linguaggio dinamicamente tipato il controllo dei tipi avviene sempre interamente a tempo di compilazione

- A) si
- B) no

ii) In una implementazione compilativa pura il compilatore riceve anche l'input del programma sorgente?

- A) si
- B) no

### **Versione 7 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Se in Java si usa unicamente il polimorfismo parametrico, allora tutti i controlli di tipo possono avvenire a tempo di compilazione?

- A) si
- B) no

ii) Gli attributi statici di una classe Java sono memorizzati nello Heap?

- A) si
- B) no

### **Versione 8 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In un programma che usa record varianti (come le union del C) il controllo dei tipi può essere fatto interamente a tempo di compilazione?

- A) si
- B) no

ii) Il primo Fortran aveva lo stack di attivazione?

- A) si
- B) no

### **Versione 9 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Esistono linguaggi fortemente e staticamente tipati dove gli identificatori non vanno necessariamente dichiarati prima dell'uso

- A) si
- B) no

ii) Il C adotta sempre la structural equivalence tra tipi?

- A) si
- B) no

### **Versione 10 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Java ha una implementazione:
  - A) compilata
  - B) interpretata
  - C) mista
  
- ii) Il primo garbage collector è stato introdotto
  - A) in Java
  - B) in C#
  - C) in Lisp

### **Versione 11 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Il C ha una implementazione:

- A) compilata
- B) interpretata
- C) mista

ii) Se A è una istanza di B e B è una istanza di C allora A è una istanza di C?

- A) si
- B) no

### **Versione 12 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Il polimorfismo che permette più controlli a tempo di compilazione è quello
  - A) per inclusione
  - B) parametrico
  
- ii) Il polimorfismo per inclusione è meno indicato di quello parametrico per la definizione di collezioni di oggetti omogenei?
  - A) si
  - B) no

### **Versione 13 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) L'ambiente associa gli identificatori a una locazione nei linguaggi:

- A) imperativi
- B) funzionali
- C) fortemente tipizzati

ii) Il primo Fortran aveva uno heap?

- A) si
- B) no

### **Versione 14 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Il polimorfismo per inclusione è meno indicato di quello parametrico per la definizione di collezioni di oggetti omogenei?

- A) si
- B) no

ii) In una implementazione compilativa pura il compilatore riceve anche l'input del programma sorgente?

- A) si
- B) no

### **Versione 15 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Se implementiamo le liste mediante template, i membri di una lista hanno tutti lo stesso tipo (o nel caso object oriented delle sottoclassi di quel tipo)?

- A) si
- B) no

ii) Il Lisp ha un garbage collector?

- A) si
- B) no

### **Versione 16 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) La dimensione degli oggetti nello heap può essere esponenziale nella altezza della gerarchia delle classi (ovvero nel numero di superclassi di una data classe)
  - A) in C
  - B) in C++
  - C) in Java
  - D) mai
  
- ii) Il primo Fortran aveva uno heap?
  - A) si
  - B) no

### **Versione 17 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Un linguaggio puramente funzionale ha i cicli while?
  - A) si
  - B) no
  
- ii) In C++ l'ereditarietà multipla può causare un consumo esponenziale di memoria?
  - A) si
  - B) no

### **Versione 18 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) ML supporta l'equivalenza di tipi
  - A) per nome (name equivalence)
  - B) strutturale (structural equivalence)
  
- ii) Nel paradigma logico una chiamata a un predicato può restituire più di un risultato?
  - A) si
  - B) no

### **Versione 1 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = y[4]$ , sapendo che x è un puntatore e che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a))$  e  $mem(env(y))$ .

### **Versione 2 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = \&y$ , sapendo che  $x$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(mem(env(a)) + 1)$ .

### **Versione 3 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(*x) = y$ , sapendo che  $x$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + 4$  e  $mem(mem(env(y)))$ .

#### **Versione 4 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[a] = *(a + z)$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(mem(env(a)) + mem(env(z)))$ .

### **Versione 5 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(a + z) = *(y)$ , sapendo che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a)) + mem(env(z))$  e  $mem(env(y))$ .

### **Versione 6 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  
 $*a = \&(y[*p])$ , sapendo che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a)) + mem(env(z))$  e  $mem(env(y))$ .

**Versione 7 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  
 $x = y$ .
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a)) + 2$  e  $env(y) + 1$ .

**Versione 8 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(*x) = y$ , sapendo che  $x$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(env(y) + mem(mem(env(p))))$ .

**Versione 9 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x = \&y$ .
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(mem(env(y)))$ .

#### **Versione 10 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(*p + 1) = *y$ , sapendo che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(env(a))$  e  $env(y) + mem(mem(env(p)))$ .

### **Versione 1 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[a] = *(*y)$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a))$  e  $mem(env(y) + 1)$ .

### **Versione 2 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[1] = *(*y)$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a))$  e  $mem(mem(env(a)))$ .

**Versione 3 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  
 $x = y$ .
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a)) + 5$  e  $env(y)$ .

#### **Versione 4 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = \&y$ , sapendo che  $x$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + 4$  e  $mem(env(y))$ .

### **Versione 5 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = y[3]$ , sapendo che  $x$  è un puntatore e che  $y$  è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(mem(env(p)))$  e  $env(y) + mem(env(a))$ .

### **Versione 6 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = y[*p]$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a)) + 5$  e  $mem(mem(env(y)))$ .

### **Versione 7 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = y$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(env(a))$  e  $env(y) + 3$ .

**Versione 8 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[a] = \&(y[a])$ , sapendo che x è un vettore e che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(env(y) + mem(env(a)))$ .

### **Versione 9 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[1] = *(*p)$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $env(y)$ .

#### **Versione 10 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast x) = \ast y$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a))$  e  $env(y) + 2$ .

### **Versione 11 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[3] = *(a + z)$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(mem(env(p)))$  e  $env(y) + 1$ .

### **Versione 1 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[a] = *(a + z)$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(mem(env(p)))$  e  $mem(env(y))$ .

**Versione 2 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(\alpha + 1) = *\alpha$ .
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $\text{env}(x)$  e  $\text{mem}(\text{env}(y) + 1)$ .

### **Versione 3 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = \&(y[2])$ , sapendo che x è un puntatore e che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(env(a))$  e  $env(y) + mem(mem(env(p)))$ .

#### **Versione 4 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = \&(y[4])$ , sapendo che x è un puntatore e che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $env(y) + mem(env(a))$ .

### **Versione 5 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  
 $*x = *(*y)$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(env(y) + mem(mem(env(p))))$ .

### **Versione 6 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast x) = \&y$ , sapendo che  $x$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a)) + 3$  e  $mem(env(y))$ .

### **Versione 7 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = y[*p]$ , sapendo che x è un puntatore e che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a))$  e  $env(y) + mem(env(a))$ .

### **Versione 8 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  
 $*x = \&y$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a)) + mem(env(z))$  e  $mem(mem(env(a)) + mem(env(z)))$ .

### **Versione 9 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = *(*y)$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $env(y) + mem(mem(env(p)))$ .

#### **Versione 10 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast x) = \ast(a + 2)$ , sapendo che  $x$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(mem(env(a)) + 5)$ .

### **Versione 11 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast x) = \ast(\ast y)$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a)) + 4$  e  $mem(env(y))$ .

### **Versione 12 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(*x) = y[*p]$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(mem(env(a)) + 1)$ .

**Versione 13 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(*x) = y$ , sapendo che  $x$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a))$  e  $mem(mem(env(y)))$ .

**Versione 14 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(\alpha + 2) = *y$ , sapendo che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(env(y))$ .

### **Versione 15 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast x) = y[a]$ , sapendo che  $x$  è un puntatore e che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(mem(env(p)))$  e  $mem(env(y))$ .

**Versione 16 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[5] = y[5]$ , sapendo che  $x$  è un vettore e che  $y$  è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $env(y)$ .

### **Versione 17 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(*x) = y[*p]$ , sapendo che x è un puntatore e che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(mem(env(p)))$  e  $mem(mem(env(y)))$ .

**Versione 18 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(*p + 5) = y[*p]$ , sapendo che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(env(y))$ .

### Versione 1 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            m();
        }
        catch( MyExc2 t ) {
        }
        catch( MyExc1 t ) {
            System.out.print(2);
            throw( new MyExc1() );
        }
        finally {
            System.out.print(3);
        }
    }
    static void m() {
        try {
            System.out.print(4);
            throw( new MyExc2() );
        }
        catch( MyExc2 e ) {
            System.out.print(5);
            throw( new MyExc1() );
        }
        catch( Error g ) {
            System.out.print(6);
            throw( new Error() );
        }
        finally {
            System.out.print(7);
        }
    }
}
```

- A) 1452Exception in thread "main" MyExc1
- B) 14566666... (ciclo infinito)
- C) Errore a tempo di compilazione

D) 145723Exception in thread "main" MyExc1

E) Nessuna delle precedenti

## Versione 2 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class C1 {
    public static void main(String [] argv) {
        try {
            p();
            System.out.print(1);
        }
        catch( Exception x ) {
            throw( new Exception() );
        }
        finally {
            throw( new MyExc3() );
        }
    }
    static void p() {
        try {
            System.out.print(2);
        }
        catch( MyExc1 w ) {
            throw( new MyExc1() );
        }
        finally {
            System.out.print(3);
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 231... (ciclo infinito)
- C) 231Exception in thread "main" MyExc3
- D) 21
- E) Nessuna delle precedenti

### Versione 3 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            n();
        }
        catch( MyExc3 y ) {
            System.out.print(2);
        }
        finally {
            System.out.print(3);
            throw( new MyExc2() );
        }
    }
    static void n() {
        try {
            throw( new MyExc3() );
        }
        catch( MyExc3 k ) {
            System.out.print(4);
        }
        catch( MyExc1 j ) {
            System.out.print(5);
        }
        catch( Exception e ) {
            System.out.print(6);
        }
        finally {
            throw( new MyExc1() );
        }
    }
}
```

- A) 1453Exception in thread "main" MyExc2
- B) Errore a tempo di compilazione
- C) 143Exception in thread "main" MyExc2
- D) 14
- E) Nessuna delle precedenti

#### Versione 4 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Error { }
public class C1 {
    public static void main(String [] argv) {
        try {
            n();
            System.out.print(1);
        }
        catch( MyExc3 k ) {
            System.out.print(2);
        }
        catch( MyExc2 i ) {
            System.out.print(3);
        }
    }
    static void n() {
        try {
            throw( new MyExc2() );
        }
        catch( MyExc1 r ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
        }
    }
}
```

- A) 4
- B) 41
- C) Errore a tempo di compilazione
- D) 341
- E) Nessuna delle precedenti

### Versione 5 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class B1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            p();
        }
        finally {
            System.out.print(2);
        }
    }
    static void p() {
        try {
        }
        catch( MyExc3 b ) {
            System.out.print(3);
        }
        catch( MyExc2 y ) {
            System.out.print(4);
        }
        catch( Exception b ) {
        }
        finally {
            System.out.print(5);
            throw( new MyExc1() );
        }
    }
}
```

- A) 1
- B) 152Exception in thread "main" MyExc1
- C) Errore a tempo di compilazione
- D) 152
- E) Nessuna delle precedenti

### Versione 6 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv) {
        try {
            m();
            System.out.print(1);
        }
        catch( Error e ) {
            System.out.print(2);
        }
        finally {
            throw( new MyExc3() );
        }
    }
    static void m() {
        try {
        }
        catch( Error t ) {
            System.out.print(3);
        }
    }
}
```

- A) 1
- B) 1Exception in thread "main" MyExc3
- C) Errore a tempo di compilazione
- D) 12
- E) Nessuna delle precedenti

### Versione 7 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Error { }
public class C1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            p();
            System.out.print(2);
        }
        catch( MyExc1 d ) {
            System.out.print(3);
        }
        finally {
            throw( new MyExc3() );
        }
    }
    static void p() {
        try {
            throw( new MyExc1() );
        }
        finally {
            System.out.print(4);
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 143Exception in thread "main" MyExc3
- C) 1432Exception in thread "main" MyExc3
- D) 142
- E) Nessuna delle precedenti

### Versione 8 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class B1 {
    public static void main(String [] argv) {
        try {
            q();
        }
        catch( MyExc2 h ) {
            System.out.print(1);
            throw( new MyExc3() );
        }
        catch( MyExc1 y ) {
            System.out.print(2);
        }
        finally {
            System.out.print(3);
            throw( new MyExc1() );
        }
    }
    static void q() {
        try {
            System.out.print(4);
            throw( new MyExc3() );
        }
        catch( MyExc3 u ) {
        }
        catch( MyExc2 b ) {
            System.out.print(5);
            throw( new MyExc2() );
        }
        catch( MyExc1 z ) {
            System.out.print(6);
        }
    }
}
```

- A) 43Exception in thread "main" MyExc1
- B) 4
- C) Errore a tempo di compilazione
- D) 432

E) Nessuna delle precedenti

### Versione 9 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            n();
            System.out.print(2);
        }
        catch( MyExc1 x ) {
        }
        catch( Error b ) {
            System.out.print(3);
        }
        finally {
            throw( new Error() );
        }
    }
    static void n() {
        try {
            System.out.print(4);
            throw( new MyExc3() );
        }
        catch( MyExc3 k ) {
            System.out.print(5);
        }
    }
}
```

- A) 1452Exception in thread "main" Error
- B) Errore a tempo di compilazione
- C) 1452
- D) 14523
- E) Nessuna delle precedenti

### Versione 10 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends MyExc2 { }
public class C1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            q();
            System.out.print(2);
        }
        catch( MyExc1 t ) {
        }
        catch( MyExc3 z ) {
        }
        catch( Error r ) {
            System.out.print(3);
        }
        finally {
            throw( new Error() );
        }
    }
    static void q() {
        try {
            throw( new MyExc1() );
        }
        catch( MyExc3 g ) {
            throw( new MyExc3() );
        }
        catch( MyExc1 h ) {
            throw( new MyExc1() );
        }
        catch( Error t ) {
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 1... (ciclo infinito)
- C) 12Exception in thread "main" Error
- D) 1
- E) Nessuna delle precedenti

### Versione 1 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class C1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            p();
        }
        finally {
            System.out.print(1);
            throw( new MyExc2() );
        }
    }
    static void p()
        throws Exception {
        try {
        }
        catch( Exception s ) {
            System.out.print(2);
            throw( new Exception() );
        }
        catch( MyExc2 u ) {
            System.out.print(3);
            throw( new MyExc1() );
        }
        finally {
            System.out.print(4);
            throw( new Exception() );
        }
    }
}
```

- A) 41Exception in thread "main" MyExc2
- B) Errore a tempo di compilazione
- C)
- D) 421Exception in thread "main" MyExc2
- E) Nessuna delle precedenti

## Versione 2 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            p();
            System.out.print(1);
        }
        finally {
            System.out.print(2);
        }
    }
    static void p()
        throws Exception {
        try {
            System.out.print(3);
            throw( new MyExc2() );
        }
        catch( Exception v ) {
            throw( new MyExc2() );
        }
        catch( MyExc3 j ) {
            throw( new MyExc1() );
        }
        finally {
            System.out.print(4);
            throw( new MyExc1() );
        }
    }
}
```

- A) 342
- B) 342Exception in thread "main" MyExc1
- C) 3... (ciclo infinito)
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 3 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            q();
            System.out.print(2);
        }
        catch( MyExc1 f ) {
            System.out.print(3);
        }
        catch( MyExc3 j ) {
        }
        finally {
            System.out.print(4);
        }
    }
    static void q()
        throws Exception {
        try {
            System.out.print(5);
            throw( new MyExc1() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 15324
- C) 153
- D) 1534
- E) Nessuna delle precedenti

### Versione 4 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class C1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            n();
            System.out.print(2);
        }
        catch( MyExc1 w ) {
        }
        catch( Exception i ) {
            System.out.print(3);
        }
        catch( MyExc2 t ) {
        }
        finally {
            System.out.print(4);
            throw( new MyExc2() );
        }
    }
    static void n()
        throws Exception {
        try {
            System.out.print(5);
            throw( new Exception() );
        }
        catch( Exception u ) {
            throw( new MyExc2() );
        }
        finally {
            System.out.print(6);
            throw( new MyExc1() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 1564Exception in thread "main" MyExc2
- C) 15

- D) 15... (ciclo infinito)
- E) Nessuna delle precedenti

## Versione 5 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            q();
            System.out.print(2);
        }
        catch( Exception i ) {
            System.out.print(3);
            throw( new MyExc3() );
        }
        finally {
            System.out.print(4);
            throw( new MyExc2() );
        }
    }
    static void q()
        throws Exception {
        try {
            System.out.print(5);
            throw( new MyExc2() );
        }
        catch( MyExc2 g ) {
            System.out.print(6);
        }
        catch( MyExc3 b ) {
        }
        catch( MyExc1 x ) {
            System.out.print(7);
        }
        finally {
            System.out.print(8);
        }
    }
}
```

- A) 1562
- B) 156824Exception in thread "main" MyExc2

- C) Errore a tempo di compilazione
- D) 156824333333... (ciclo infinito)
- E) Nessuna delle precedenti

### Versione 6 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            m();
            System.out.print(1);
        }
        catch( Exception z ) {
            System.out.print(2);
        }
        catch( MyExc3 y ) {
            throw( new MyExc3() );
        }
    }
    static void m()
        throws Exception {
        try {
            throw( new Exception() );
        }
        finally {
            throw( new MyExc2() );
        }
    }
}
```

- A) 2Exception in thread "main" java.lang.Exception
- B) 2
- C) Errore a tempo di compilazione
- D) Exception in thread "main" MyExc2
- E) Nessuna delle precedenti

### Versione 7 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            p();
            System.out.print(2);
        }
        catch( Exception r ) {
            System.out.print(3);
        }
        catch( MyExc1 j ) {
            System.out.print(4);
        }
    }
    static void p()
        throws Exception {
        try {
            throw( new MyExc3() );
        }
        catch( Exception f ) {
            System.out.print(5);
        }
        catch( MyExc1 j ) {
        }
        catch( MyExc3 w ) {
        }
        finally {
            System.out.print(6);
            throw( new Exception() );
        }
    }
}
```

- A) 1563
- B) Errore a tempo di compilazione
- C) 12
- D) 15632

E) Nessuna delle precedenti

### Versione 8 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            q();
        }
        catch( Exception j ) {
        }
        catch( MyExc1 w ) {
            System.out.print(1);
        }
        finally {
            System.out.print(2);
        }
    }
    static void q()
        throws Exception {
        try {
            throw( new MyExc1() );
        }
    }
}
```

- A) 1
- B) 2
- C) 12
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 9 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class C1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            q();
        }
        catch( MyExc3 e ) {
            System.out.print(2);
        }
        catch( Exception d ) {
            System.out.print(3);
        }
        catch( MyExc1 s ) {
            System.out.print(4);
            throw( new MyExc1() );
        }
        finally {
            System.out.print(5);
        }
    }
    static void q()
        throws Exception {
        try {
            throw( new MyExc1() );
        }
        catch( MyExc2 z ) {
            System.out.print(6);
        }
        catch( MyExc3 t ) {
            System.out.print(7);
        }
        catch( MyExc1 z ) {
        }
        finally {
            throw( new MyExc1() );
        }
    }
}
```

- A) 145Exception in thread "main" MyExc1
- B) Errore a tempo di compilazione
- C) 135
- D) 1
- E) Nessuna delle precedenti

### Versione 10 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            p();
            System.out.print(1);
        }
        catch( MyExc3 i ) {
            System.out.print(2);
            throw( new Exception() );
        }
        catch( MyExc2 x ) {
        }
        finally {
            System.out.print(3);
            throw( new MyExc3() );
        }
    }
    static void p()
        throws Exception {
        try {
            throw( new MyExc1() );
        }
        catch( Exception b ) {
            System.out.print(4);
        }
        catch( MyExc3 z ) {
            System.out.print(5);
            throw( new MyExc3() );
        }
        finally {
            System.out.print(6);
            throw( new MyExc3() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 464132Exception in thread "main" java.lang.Exception

- C) 4623Exception in thread "main" MyExc3
- D) 623Exception in thread "main" MyExc3
- E) Nessuna delle precedenti

### Versione 11 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            p();
        }
        catch( MyExc2 f ) {
        }
        catch( MyExc3 w ) {
            System.out.print(2);
        }
        catch( Exception a ) {
        }
        finally {
            System.out.print(3);
            throw( new Exception() );
        }
    }
    static void p()
        throws Exception {
        try {
            throw( new MyExc3() );
        }
        catch( Exception e ) {
            System.out.print(4);
        }
        catch( MyExc2 e ) {
            System.out.print(5);
        }
        catch( MyExc1 g ) {
        }
    }
}
```

- A) 123Exception in thread "main" java.lang.Exception
- B) Errore a tempo di compilazione
- C) 143Exception in thread "main" java.lang.Exception

D) 12

E) Nessuna delle precedenti

### Versione 1 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            n();
        }
        catch( MyExc1 t ) {
            System.out.print(2);
            throw( new Error() );
        }
        catch( MyExc3 g ) {
            throw( new MyExc2() );
        }
        catch( Error g ) {
        }
        finally {
            System.out.print(3);
        }
    }
    static void n() {
        try {
            throw( new MyExc3() );
        }
        finally {
            System.out.print(4);
        }
    }
}
```

- A) 143
- B) Errore a tempo di compilazione
- C) 14
- D) 143Exception in thread "main" MyExc2
- E) Nessuna delle precedenti

## Versione 2 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            n();
        }
        catch( MyExc1 h ) {
            System.out.print(2);
            throw( new Exception() );
        }
        finally {
            throw( new MyExc1() );
        }
    }
    static void n() {
        try {
        }
        catch( MyExc3 c ) {
        }
        finally {
            System.out.print(3);
        }
    }
}
```

- A) 13Exception in thread "main" MyExc1
- B) 132Exception in thread "main" java.lang.Exception
- C) Errore a tempo di compilazione
- D) 1
- E) Nessuna delle precedenti

### Versione 3 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class D1 {
    public static void main(String [] argv) {
        try {
            q();
        }
        catch( Exception j ) {
            throw( new Exception() );
        }
    }
    static void q() {
        try {
            System.out.print(1);
            throw( new MyExc3() );
        }
        catch( Exception j ) {
            throw( new Exception() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 1... (ciclo infinito)
- C) 1Exception in thread "main" MyExc3
- D) 1Exception in thread "main" java.lang.Exception
- E) Nessuna delle precedenti

### Versione 4 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends MyExc2 { }
public class D1 {
    public static void main(String [] argv) {
        try {
            q();
            System.out.print(1);
        }
        catch( MyExc1 z ) {
        }
        catch( MyExc3 b ) {
        }
        catch( MyExc2 u ) {
        }
        finally {
            System.out.print(2);
            throw( new MyExc3() );
        }
    }
    static void q() {
        try {
            System.out.print(3);
        }
        catch( MyExc1 v ) {
            System.out.print(4);
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 312Exception in thread "main" MyExc3
- C) 312
- D) 31
- E) Nessuna delle precedenti

### Versione 5 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv) {
        try {
            m();
        }
        catch( MyExc1 z ) {
            throw( new MyExc3() );
        }
        finally {
            System.out.print(1);
        }
    }
    static void m() {
        try {
            System.out.print(2);
            throw( new Exception() );
        }
        catch( MyExc2 a ) {
            System.out.print(3);
        }
        catch( Exception u ) {
            throw( new MyExc3() );
        }
        finally {
            System.out.print(4);
        }
    }
}
```

- A) 21
- B) Errore a tempo di compilazione
- C) 2341
- D) 241Exception in thread "main" MyExc3
- E) Nessuna delle precedenti

### Versione 6 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class C1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            q();
        }
        catch( Error j ) {
            throw( new MyExc1() );
        }
    }
    static void q() {
        try {
            throw( new Error() );
        }
        finally {
            System.out.print(2);
        }
    }
}
```

- A) 12
- B) Errore a tempo di compilazione
- C) 12Exception in thread "main" MyExc1
- D) 12... (ciclo infinito)
- E) Nessuna delle precedenti

### Versione 7 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class A1 {
    public static void main(String [] argv) {
        try {
            q();
        }
        catch( MyExc3 i ) {
            System.out.print(1);
        }
    }
    static void q() {
        try {
            throw( new MyExc1() );
        }
        catch( Exception a ) {
            System.out.print(2);
        }
        finally {
            throw( new Exception() );
        }
    }
}
```

- A) 22
- B) Errore a tempo di compilazione
- C) Exception in thread "main" java.lang.Exception
- D) 2Exception in thread "main" java.lang.Exception
- E) Nessuna delle precedenti

### Versione 8 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class D1 {
    public static void main(String [] argv) {
        try {
            q();
            System.out.print(1);
        }
        finally {
            System.out.print(2);
        }
    }
    static void q() {
        try {
        }
        catch( Exception i ) {
            System.out.print(3);
        }
        finally {
            System.out.print(4);
            throw( new MyExc3() );
        }
    }
}
```

- A) 4312
- B) 1
- C) Errore a tempo di compilazione
- D) 42Exception in thread "main" MyExc3
- E) Nessuna delle precedenti

### Versione 9 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class C1 {
    public static void main(String [] argv) {
        try {
            n();
            System.out.print(1);
        }
        catch( MyExc3 v ) {
        }
        catch( MyExc1 c ) {
        }
        catch( Exception x ) {
            System.out.print(2);
            throw( new MyExc2() );
        }
        finally {
            throw( new Exception() );
        }
    }
    static void n() {
        try {
            throw( new MyExc2() );
        }
        catch( MyExc2 b ) {
            System.out.print(3);
            throw( new MyExc3() );
        }
        catch( MyExc3 d ) {
        }
        catch( Exception e ) {
        }
        finally {
            throw( new MyExc1() );
        }
    }
}
```

- A) 312
- B) 3
- C) 3Exception in thread "main" java.lang.Exception

D) Errore a tempo di compilazione

E) Nessuna delle precedenti

### Versione 10 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class A1 {
    public static void main(String [] argv) {
        try {
            n();
            System.out.print(1);
        }
        finally {
            System.out.print(2);
        }
    }
    static void n() {
        try {
            throw( new Exception() );
        }
        catch( Exception a ) {
            System.out.print(3);
            throw( new Exception() );
        }
    }
}
```

- A) 32
- B) Errore a tempo di compilazione
- C) 333333... (ciclo infinito)
- D) 32Exception in thread "main" java.lang.Exception
- E) Nessuna delle precedenti

### Versione 11 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            q();
            System.out.print(2);
        }
        finally {
            System.out.print(3);
            throw( new Error() );
        }
    }
    static void q() {
        try {
            System.out.print(4);
            throw( new Error() );
        }
        catch( MyExc1 y ) {
            System.out.print(5);
        }
        catch( Error y ) {
            throw( new Error() );
        }
        finally {
            System.out.print(6);
        }
    }
}
```

- A) 1463Exception in thread "main" Error
- B) 143Exception in thread "main" Error
- C) 14... (ciclo infinito)
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 12 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends Error { }
public class C1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            q();
        }
        catch( MyExc1 r ) {
            System.out.print(2);
        }
        catch( MyExc3 a ) {
            System.out.print(3);
        }
        catch( MyExc2 e ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
        }
    }
    static void q() {
        try {
            System.out.print(6);
            throw( new MyExc2() );
        }
        catch( Error d ) {
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 165
- C) 164
- D) 1645
- E) Nessuna delle precedenti

### Versione 13 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends Error { }
public class A1 {
    public static void main(String [] argv) {
        try {
            m();
        }
        catch( MyExc2 c ) {
        }
        catch( MyExc3 j ) {
            throw( new MyExc3() );
        }
    }
    static void m() {
        try {
            System.out.print(1);
        }
        catch( MyExc3 h ) {
        }
        catch( MyExc1 s ) {
        }
        finally {
            System.out.print(2);
            throw( new MyExc1() );
        }
    }
}
```

- A) 12
- B) 1
- C) 12Exception in thread "main" MyExc1
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 14 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends Error { }
public class D1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            q();
            System.out.print(2);
        }
        catch( Error v ) {
        }
        finally {
            System.out.print(3);
            throw( new Error() );
        }
    }
    static void q() {
        try {
            throw( new MyExc3() );
        }
        catch( MyExc2 t ) {
            System.out.print(4);
        }
        catch( Error j ) {
            throw( new MyExc1() );
        }
        finally {
            System.out.print(5);
        }
    }
}
```

- A) 1523Exception in thread "main" Error
- B) 152
- C) Errore a tempo di compilazione
- D) 1... (ciclo infinito)
- E) Nessuna delle precedenti

### Versione 15 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv) {
        try {
            n();
            System.out.print(1);
        }
        catch( MyExc1 a ) {
            System.out.print(2);
        }
        catch( MyExc3 v ) {
        }
        catch( Error d ) {
            throw( new MyExc3() );
        }
        finally {
            System.out.print(3);
        }
    }
    static void n() {
        try {
            System.out.print(4);
            throw( new MyExc3() );
        }
        catch( MyExc2 z ) {
            System.out.print(5);
        }
        catch( MyExc1 w ) {
        }
        finally {
            System.out.print(6);
        }
    }
}
```

- A) 463
- B) 461
- C) 45613
- D) Errore a tempo di compilazione

E) Nessuna delle precedenti

### Versione 16 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends Error { }
public class A1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            m();
            System.out.print(2);
        }
        finally {
            throw( new MyExc1() );
        }
    }
    static void m() {
        try {
            System.out.print(3);
            throw( new MyExc3() );
        }
        catch( MyExc1 h ) {
        }
        catch( MyExc3 a ) {
            System.out.print(4);
        }
        catch( Error c ) {
            System.out.print(5);
        }
        finally {
            throw( new MyExc3() );
        }
    }
}
```

- A) 134Exception in thread "main" MyExc1
- B) Errore a tempo di compilazione
- C) 13442Exception in thread "main" MyExc1
- D) 1342
- E) Nessuna delle precedenti

### Versione 17 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class B1 {
    public static void main(String [] argv) {
        try {
            q();
        }
        catch( MyExc2 d ) {
            System.out.print(1);
            throw( new Exception() );
        }
        catch( Exception b ) {
        }
    }
    static void q() {
        try {
            System.out.print(2);
            throw( new MyExc3() );
        }
        catch( Exception f ) {
            throw( new MyExc2() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 2... (ciclo infinito)
- C) 2Exception in thread "main" MyExc3
- D) 21Exception in thread "main" java.lang.Exception
- E) Nessuna delle precedenti

### Versione 18 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends MyExc2 { }
public class D1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            n();
        }
        catch( Error x ) {
            System.out.print(2);
            throw( new MyExc3() );
        }
        finally {
            System.out.print(3);
            throw( new MyExc2() );
        }
    }
    static void n() {
        try {
            throw( new MyExc2() );
        }
        finally {
            System.out.print(4);
        }
    }
}
```

- A) 1423Exception in thread "main" MyExc2
- B) 14222222... (ciclo infinito)
- C) Errore a tempo di compilazione
- D) 14
- E) Nessuna delle precedenti

### **Versione 1 dell'esercizio 6**

Date le seguenti classi:

```
class A {  
    Integer i = 6;  
}  
class B extends A {  
}
```

Quale delle seguenti proposte di cambiamento del codice precedente è la migliore rispetto all'incapsulamento?

- A) Le classi sono perfettamente incapsulate.
- B) Definire **private** la variabile i e aggiungere due metodi, uno che ne modifichi ed uno che ne restituisca il valore.
- C) Definire **final** la variabile i.
- D) Definire **protected** la variabile i e aggiungere due metodi, uno che ne modifichi ed uno che ne restituisca il valore.
- E) Nessuna delle precedenti.

## Versione 2 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public class Person {
    static String name, surname;

    Person(String n, String s) {
        name = n;
        surname = s;
    }

    public static String id() {
        return name + " ";
    }
}

class Employee extends Person {
    Employee(String n, String s) {super(n,s);}

    public static String id() {
        return name + " " + surname + " ";
    }
}

class Main {
    public static void main(String[] args) {
        Person a = new Person("Ann","Taylor");
        Person e = new Employee("Eva","Smith");
        Employee j = new Employee("Jim","Brown");
        System.out.print(a.id());
        System.out.print(e.id());
        System.out.print(j.id());
    }
}
```

- A) Ann Eva Jim Brown
- B) Ann Eva Smith Jim Brown
- C) Jim Jim Jim Brown
- D) Errore di compilazione.
- E) Errore di esecuzione.

### Versione 3 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public abstract class X {  
    X(long l){  
        System.out.print("X");  
    }  
    abstract void z();  
}  
  
class Y extends X {  
    void z(){  
        Y(){  
            System.out.print("Y");  
        }  
    }  
  
    public static void main(String[] args){  
        X x = new Y();  
    }  
}
```

- A) X
- B) Y
- C) Errore a tempo di compilazione.
- D) Errore a tempo di esecuzione.
- E) Nessuna delle precedenti.

#### **Versione 4 dell'esercizio 6**

Qual'è tra le seguenti dichiarazioni non produce un errore di compilazione?

```
public interface I {  
1.     byte b;  
2.     private final int i = 1;  
3.     public static final boolean b;  
4.     static float f = 3;  
5.     protected double m();  
}
```

- A) Quella alla linea 1.
- B) Quella alla linea 2.
- C) Quella alla linea 3.
- D) Quella alla linea 4.
- E) Quella alla linea 5.

## Versione 5 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
interface A {
    String name = "A";
    void m();
}

abstract class B implements A {
    String name = "B";
    abstract B g();
}

class C extends B {
    String name = "C";
    void m() {
        System.out.print(
            ((A)this).name +
            ((B)this).name +
            ((C)this).name);
    }
    B g() {return this;}
    public static void main(String[] args){
        new C().m();
    }
}
```

- A) ABC
- B) CCC
- C) Errore a tempo di compilazione.
- D) Errore a tempo di esecuzione.
- E) Risultato diverso dai precedenti.

## Versione 6 dell'esercizio 6

Cosa succede compilando e, se possibile, eseguendo questo codice

```
class C {  
    static int x;  
    C() { x=1; }  
    public static void m() {  
        System.out.println("x di C vale "+x);  
    }  
}  
  
class D extends C {  
    int x = 2;  
    public void m() {  
        System.out.println("x di D vale "+x);  
    }  
}  
  
public class Test {  
    int x = 3;  
    public static void main(String[] a) {  
        C y = new D();  
        y.m();  
    }  
}
```

- A) Errore a tempo di compilazione
- B) Stampa x di C vale 1
- C) Stampa x di C vale 2
- D) Stampa x di D vale 1
- E) Stampa x di D vale 2

### **Versione 7 dell'esercizio 6**

Date le seguenti classi:

```
class A {  
    Integer i = 6;  
}  
class B extends A {  
}
```

Quale delle seguenti proposte di cambiamento del codice precedente è la migliore rispetto all'incapsulamento?

- A) Le classi sono perfettamente incapsulate.
- B) Definire **private** la variabile i e aggiungere due metodi, uno che ne modifichi ed uno che ne restituisca il valore.
- C) Definire **final** la variabile i.
- D) Definire **protected** la variabile i e aggiungere due metodi, uno che ne modifichi ed uno che ne restituisca il valore.
- E) Nessuna delle precedenti.

### Versione 8 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
interface I {
    void f();
}

abstract class A implements I {
    String s = "A";
}

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

- A) IAB
- B) BBB
- C) Errore a tempo di compilazione.
- D) Errore a tempo di esecuzione.
- E) Risultato diverso dai precedenti.

### Versione 9 dell'esercizio 6

Qual è l'output di questo codice?

```
interface FiguraGeometrica {  
    static int dimensioni = 10;  
    void print();  
}  
  
class Cono implements FiguraGeometrica {  
  
    static final int numLati = 3;  
  
    public void print() {  
        System.out.println("Cono: "  
            + dimensioni + " - " +  
            numLati + " lati per faccia.");  
    }  
  
    public static void main(String [] argv) {  
        Cono c1 = new Cono();  
        FiguraGeometrica f1 =c1;  
        c1.print();  
    }  
}
```

- A) Errore a tempo di compilazione
- B) Errore a tempo di esecuzione
- C) Cono: 8 - 3 lati per faccia
- D) Cono: 3 - 3 lati per faccia
- E) Nessuna delle precedenti

### Versione 10 dell'esercizio 6

Date le seguenti classi:

```
class A {  
    Integer i = 6;  
}  
class B extends A {  
}
```

Quale delle seguenti proposte di cambiamento del codice precedente è la migliore rispetto all'incapsulamento?

- A) Le classi sono perfettamente incapsulate.
- B) Definire **private** la variabile i e aggiungere due metodi, uno che ne modifichi ed uno che ne restituisca il valore.
- C) Definire **final** la variabile i.
- D) Definire **protected** la variabile i e aggiungere due metodi, uno che ne modifichi ed uno che ne restituisca il valore.
- E) Nessuna delle precedenti.

## Versione 1 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     int y=1;  
  
4:     class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int m() {  
8:             return x+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Main {  
16:     public static void main(String[] a){  
17:         C.D d = new C().new D();  
18:         System.out.println(d.m());  
19:     }  
}
```

- A) Errore di compilazione alla linea 1
- B) Errore di compilazione alla linea 8
- C) Errore di compilazione alla linea 16
- D) Stampa 3
- E) Stampa 4
- F) Stampa 5
- G) Stampa 6

## Versione 2 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     private int y=1;  
  
4:     static class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int m() {  
8:             return C.this.x+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Main {  
16:     public static void main(String[] a){  
17:         C.D d = new C.D();  
18:         System.out.println(d.m());  
19:     }  
}
```

- A) Errore di compilazione alla linea 1
- B) Errore di compilazione alla linea 8
- C) Errore di compilazione alla linea 16
- D) Stampa 3
- E) Stampa 4
- F) Stampa 5
- G) Stampa 6

### Versione 3 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     int y=1;  
  
4:     private static class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int m() {  
8:             return C.this.x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Main {  
16:     public static void main(String[] a){  
17:         C.D d = new C().new D();  
18:         System.out.println(d.m());  
19:     }  
}
```

- A) Errore di compilazione alla linea 1
- B) Errore di compilazione alla linea 8
- C) Errore di compilazione alla linea 16
- D) Stampa 3
- E) Stampa 4
- F) Stampa 5
- G) Stampa 6

#### **Versione 4 dell'esercizio 6**

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     private int y=1;  
  
4:     static class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int m() {  
8:             return C.this.x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Main {  
16:     public static void main(String[] a){  
17:         C.D d = new C().new D();  
18:         System.out.println(d.m());  
19:     }  
}
```

- A) Errore di compilazione alla linea 1
- B) Errore di compilazione alla linea 8
- C) Errore di compilazione alla linea 16
- D) Stampa 3
- E) Stampa 4
- F) Stampa 5
- G) Stampa 6

## Versione 5 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     private int y=1;  
  
4:     class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int m() {  
8:             return x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14: public class Main {  
15:     public static void main(String[] a){  
16:         C.D d = new D();  
17:         System.out.println(d.m());  
18:     }  
19: }
```

- A) Errore di compilazione alla linea 1
- B) Errore di compilazione alla linea 8
- C) Errore di compilazione alla linea 16
- D) Stampa 3
- E) Stampa 4
- F) Stampa 5
- G) Stampa 6

## Versione 6 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     private int y=1;  
  
4:     class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int m() {  
8:             return C.this.x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Main {  
16:     public static void main(String[] a){  
17:         C.D d = new C().new D();  
18:         System.out.println(d.m());  
19:     }  
}
```

- A) Errore di compilazione alla linea 1
- B) Errore di compilazione alla linea 8
- C) Errore di compilazione alla linea 16
- D) Stampa 3
- E) Stampa 4
- F) Stampa 5
- G) Stampa 6

### Versione 7 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     int y=1;  
  
4:     class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int m() {  
8:             return x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Main {  
16:     public static void main(String[] a){  
17:         C.D d = new C().new D();  
18:         System.out.println(d.m());  
19:     }  
}
```

- A) Errore di compilazione alla linea 1
- B) Errore di compilazione alla linea 8
- C) Errore di compilazione alla linea 16
- D) Stampa 3
- E) Stampa 4
- F) Stampa 5
- G) Stampa 6

### Versione 8 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: private class C {  
2:     int x;  
3:     private int y=1;  
  
4:     private class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int m() {  
8:             return C.this.x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14: public class Main {  
15:     public static void main(String[] a){  
16:         C.D d = new C().new D();  
17:         System.out.println(d.m());  
18:     }  
19: }
```

- A) Errore di compilazione alla linea 1
- B) Errore di compilazione alla linea 8
- C) Errore di compilazione alla linea 16
- D) Stampa 3
- E) Stampa 4
- F) Stampa 5
- G) Stampa 6

### Versione 9 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     private int y=1;  
  
4:     static class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int m() {  
8:             return x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Main {  
16:     public static void main(String[] a){  
17:         C.D d = new C.D();  
18:         System.out.println(d.m());  
19:     }  
}
```

- A) Errore di compilazione alla linea 1
- B) Errore di compilazione alla linea 8
- C) Errore di compilazione alla linea 16
- D) Stampa 3
- E) Stampa 4
- F) Stampa 5
- G) Stampa 6

### Versione 10 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     private int y=1;  
  
4:     static class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int m() {  
8:             return C.x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Main {  
16:     public static void main(String[] a){  
17:         C.D d = new C.D();  
18:         System.out.println(d.m());  
19:     }  
}
```

- A) Errore di compilazione alla linea 1
- B) Errore di compilazione alla linea 8
- C) Errore di compilazione alla linea 16
- D) Stampa 3
- E) Stampa 4
- F) Stampa 5
- G) Stampa 6

### Versione 11 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     int y=1;  
  
4:     static class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int m() {  
8:             return x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Main {  
16:     public static void main(String[] a){  
17:         C.D d = new C.D();  
18:         System.out.println(d.m());  
19:     }  
}
```

- A) Errore di compilazione alla linea 1
- B) Errore di compilazione alla linea 8
- C) Errore di compilazione alla linea 16
- D) Stampa 3
- E) Stampa 4
- F) Stampa 5
- G) Stampa 6

## Versione 1 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     int y=1;  
  
4:     public class D extends C {  
5:         int x=2;  
6:         int z=3;  
7:         public int p() {  
8:             return x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C d = new C().new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

## Versione 2 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     private int y=1;  
  
4:     public class D extends C {  
5:         int x=2;  
6:         int z=3;  
7:         public int p() {  
8:             return C.this.x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C.D d = c.new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

### Versione 3 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     public int y=1;  
  
4:     public class D extends C {  
5:         int x=2;  
6:         int z=3;  
7:         public int p() {  
8:             return C.this.x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C d = new C().new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

#### Versione 4 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     protected int y=1;  
  
4:     public class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int q() {  
8:             return C.this.x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C d = new C().new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

## Versione 5 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     int y=1;  
  
4:     public class D extends C {  
5:         int x=2;  
6:         int z=3;  
7:         public int p() {  
8:             return C.this.x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C.D d = new C().new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

## Versione 6 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     int y=1;  
  
4:     public class D extends C {  
5:         int x=2;  
6:         int z=3;  
7:         public int p() {  
8:             return x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C.D d = new C().new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

## Versione 7 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     protected int y=1;  
  
4:     public class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int p() {  
8:             return C.this.x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C.D d = new C().new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

## Versione 8 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     public int y=1;  
  
4:     public class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int m() {  
8:             return x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14: public class Test {  
15:     public static void main(String[] arg){  
16:         C c = new C();  
17:         C.D d = new C().new D();  
18:         System.out.print(c.m());  
19:         System.out.println(d.m());  
20:     }  
21: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

### Versione 9 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     protected int y=1;  
  
4:     public class D extends C {  
5:         int x=2;  
6:         int z=3;  
7:         public int q() {  
8:             return x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C d = c.new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

### Versione 10 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     int y=1;  
  
4:     public class D extends C {  
5:         int x=2;  
6:         int z=3;  
7:         public int p() {  
8:             return x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C d = c.new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

### Versione 11 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     public int y=1;  
  
4:     public class D extends C {  
5:         int x=2;  
6:         int z=3;  
7:         public int p() {  
8:             return C.this.x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C d = new C().new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

### Versione 12 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     public int y=1;  
  
4:     public class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int p() {  
8:             return C.this.x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C d = new C().new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

### Versione 13 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     private int y=1;  
  
4:     public class D extends C {  
5:         int x=2;  
6:         int z=3;  
7:         public int q() {  
8:             return x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C.D d = new C().new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

### Versione 14 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     private int y=1;  
  
4:     public class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int q() {  
8:             return C.this.x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C d = c.new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

### Versione 15 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     protected int y=1;  
  
4:     public class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int q() {  
8:             return C.this.x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C.D d = c.new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

### Versione 16 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     private int y=1;  
  
4:     public class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int p() {  
8:             return x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C.D d = new C().new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

### Versione 17 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     private int y=1;  
  
4:     public class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int p() {  
8:             return C.this.x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C d = c.new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

### Versione 18 dell'esercizio 6

Qual è il comportamento del seguente programma? In caso di errori indicarli tutti.

```
1: class C {  
2:     int x;  
3:     public int y=1;  
  
4:     public class D {  
5:         int x=2;  
6:         int z=3;  
7:         public int q() {  
8:             return x+y+z;  
9:         }  
10:    }  
  
11:    public int m() {  
12:        return x+y;  
13:    }  
14:}  
  
15: public class Test {  
16:     public static void main(String[] arg){  
17:         C c = new C();  
18:         C d = c.new D();  
19:         System.out.print(c.m());  
20:         System.out.println(d.m());  
21:     }  
22: }
```

- A) Errore di compilazione alla linea 8
- B) Errore di compilazione alla linea 17
- C) Errore di compilazione alla linea 18
- D) Stampa 11
- E) Stampa 14
- F) Stampa 16

**Versione 1 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML hd che restituisce la testa di una lista.

**Versione 2 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn (x,L) => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

### **Versione 3 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn x => fn L => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

**Versione 4 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML hd che restituisce la testa di una lista.

### **Versione 5 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML List.nth che restituisce l'i-esimo elemento di una lista.

**Versione 6 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML tl che restituisce la coda di una lista.

### **Versione 7 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML List.nth che restituisce l'i-esimo elemento di una lista.

**Versione 8 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML hd che restituisce la testa di una lista.

**Versione 9 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML hd che restituisce la testa di una lista.

### **Versione 10 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML length che calcola la lunghezza di una lista.

### **Versione 1 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML List.nth che restituisce l'i-esimo elemento di una lista.

**Versione 2 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML null che verifica se una lista è vuota.

### **Versione 3 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fun f (x,[]) = 0
| f (x,y::z) = x=y orelse f (x,z);
```

segnalando eventuali errori.

#### **Versione 4 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML List.nth che restituisce l'i-esimo elemento di una lista.

**Versione 5 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML null che verifica se una lista è vuota.

### **Versione 6 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn (x,L) => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

**Versione 7 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML hd che restituisce la testa di una lista.

### **Versione 8 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fun f (x,[]) = 0
| f (x,y::z) = x=y orelse f (x,z);
```

segnalando eventuali errori.

### **Versione 9 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fun f x [] = 0
| f x (y::z) = x=y orelse f x z;
```

segnalando eventuali errori.

### **Versione 10 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fun f x [] = false
  | f x (y::z) = x=y orelse f x z;
```

segnalando eventuali errori.

### **Versione 11 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn x => fn L => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

### **Versione 1 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fun f x [] = false
  | f x (y::z) = x=y orelse f x z;
```

segnalando eventuali errori.

**Versione 2 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML null che verifica se una lista è vuota.

**Versione 3 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML null che verifica se una lista è vuota.

#### **Versione 4 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML length che calcola la lunghezza di una lista.

### **Versione 5 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn (x,L) => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

### **Versione 6 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fun f (x,[]) = false
  | f (x,y::z) = x=y orelse f (x,z);
```

segnalando eventuali errori.

**Versione 7 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML null che verifica se una lista è vuota.

### **Versione 8 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML List.nth che restituisce l'i-esimo elemento di una lista.

### **Versione 9 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn (x,L) => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

### **Versione 10 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn (x,L) => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

### **Versione 11 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn (x,L) => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

### **Versione 12** dell'esercizio sui tipi di ML

Scrivere il tipo della funzione ML:

```
fn x => fn L => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

### **Versione 13 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fun f (x,[]) = false  
| f (x,y::z) = x=y orelse f (x,z);
```

segnalando eventuali errori.

### **Versione 14 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fun f (x,[]) = 0
| f (x,y::z) = x=y orelse f (x,z);
```

segnalando eventuali errori.

### **Versione 15 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML length che calcola la lunghezza di una lista.

### **Versione 16 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn x => fn L => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

### **Versione 17 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fun f (x,[]) = 0
| f (x,y::z) = x=y orelse f (x,z);
```

segnalando eventuali errori.

### **Versione 18 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn (x,L) => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

### Passaggio parametri versione n. 1

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN per copia

```
program p1
int q; int r; int s; int t;
procedure p2([MODE] int q,[IN x copia] int t)
int r;
procedure p3([IN OUT x copia] int q)
int t;
BEGIN
t=s;
if s>50 then q=r else q=s;
s=2;
r=s*4;
p4(t);
write(q,r,s,t);
END

procedure p4([IN OUT x copia] int r)
int q;
BEGIN
q=r;
r=3;
t=1;
s=1;
write(q,r,s,t);
END

BEGIN
r=4;
q=t;
t=r+4;
s=q;
p3(r);
write(q,r,s,t);
END

BEGIN
q=3;
r=3;
s=2;
t=1;
p2(s, t);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 2

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int a; int b; int c; int d;
procedure p2([MODE] int c)
int a; int d;
procedure p3([IN OUT x copia] int c,[IN OUT x copia] int d)
int a;
BEGIN
a=3;
c=d-4;
d=c;
b=b*1;
p4(c, d);
write(a,b,c,d);
END

BEGIN
a=3;
d=1;
c=4;
b=d*3;
p3(b, d);
write(a,b,c,d);
END

procedure p4([IN OUT x rif] int a,[IN OUT x rif] int c)
int b;
BEGIN
b=a;
a=3;
c=b+1;
d=c*4;
write(a,b,c,d);
END

BEGIN
a=4;
b=4;
c=1;
d=2;
p2(d);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 3

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN OUT per copia

```
program p1
int x; int y; int z; int t;
procedure p2([IN x copia] int t,[MODE] int z)
int y;
procedure p3([IN OUT x rif] int z)
int y; int t;
BEGIN
y=z-3;
t=x+4;
if z>5 then z=y else z=2;
x=1;
write(x,y,z,t);
END

BEGIN
y=4;
t=3;
if t>2 then z=4 else z=x-1;
x=3;
p3(z);
write(x,y,z,t);
END

procedure p4([IN OUT x rif] int y)
int t; int x;
BEGIN
t=y-2;
x=z;
y=y;
z=x;
p2(x, z);
write(x,y,z,t);
END

BEGIN
x=4;
y=0;
z=4;
t=4;
p4(y);
write(x,y,z,t);
END
```

#### Passaggio parametri versione n. 4

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per riferimento

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x rif] int t,[MODE] int z)
int y;
procedure p3([IN OUT x copia] int y)
int z; int t;
BEGIN
z=y;
t=x;
y=x-3;
x=3;
p4(z);
write(x,y,z,t);
END

procedure p4([IN x copia] int y)
int x; int z;
BEGIN
if t<5 then x=3 else x=4;
z=4;
y=x*1;
t=z;
write(x,y,z,t);
END

BEGIN
y=3;
t=2;
z=x+4;
x=2;
p3(z);
write(x,y,z,t);
END

BEGIN
x=0;
y=4;
z=3;
t=1;
p2(z, y);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 5

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN per copia

```
program p1
int x; int y; int z; int t;
procedure p2([MODE] int x,[IN OUT x rif] int t)
int y;
BEGIN
y=2;
if z=7 then x=z else x=y;
t=x;
z=4;
p3(y);
write(x,y,z,t);
END

procedure p3([IN OUT x rif] int x)
int z; int y;
procedure p4([IN x copia] int y)
int z; int x;
BEGIN
z=4;
x=1;
y=2;
t=y;
write(x,y,z,t);
END

BEGIN
z=4;
y=1;
if z>9 then x=z-2 else x=2;
t=t-1;
p4(x);
write(x,y,z,t);
END

BEGIN
x=4;
y=0;
z=1;
t=0;
p2(t, x);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 6

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x copia] int t,[IN x copia] int y)
int x;
procedure p3([IN OUT x copia] int y,[IN x copia] int z)
int t;
BEGIN
t=1;
y=3;
z=4;
if t<9 then x=4 else x=3;
write(x,y,z,t);
END

procedure p4([MODE] int t,[IN x copia] int x)
int z;
BEGIN
z=4;
t=3;
x=1;
y=x;
p3(z, y);
write(x,y,z,t);
END

BEGIN
if z>2 then x=3 else x=1;
t=z*4;
if y=3 then y=1 else y=1;
z=3;
p4(t, x);
write(x,y,z,t);
END

BEGIN
x=1;
y=4;
z=3;
t=4;
p2(t, y);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 7

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x rif] int y)
int x; int t;
BEGIN
x=z+4;
t=2;
if t>3 then y=3 else y=2;
z=2;
write(x,y,z,t);
END

procedure p3([IN OUT x copia] int z)
int y; int t;
procedure p4([MODE] int x,[IN x copia] int z)
int y;
BEGIN
y=2;
x=4;
z=2;
t=2;
p2(z);
write(x,y,z,t);
END

BEGIN
y=z;
t=x;
z=2;
x=1;
p4(z, t);
write(x,y,z,t);
END

BEGIN
x=1;
y=3;
z=0;
t=2;
p3(x);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 8

- Scoping dinamico, MODE = IN per copia
- Scoping statico, MODE = OUT per copia

```
program p1
int x; int y; int z; int t;
procedure p2([MODE] int z)
int t; int x;
BEGIN
t=z;
x=2;
z=3;
y=y-3;
p3(x);
write(x,y,z,t);
END

procedure p3([IN OUT x rif] int x)
int y; int z;
procedure p4([IN OUT x copia] int t)
int x;
BEGIN
x=z;
t=x+2;
z=3;
y=3;
write(x,y,z,t);
END

BEGIN
y=1;
z=t;
x=3;
t=3;
p4(x);
write(x,y,z,t);
END

BEGIN
x=0;
y=0;
z=3;
t=1;
p2(x);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 9

- Scoping dinamico, MODE = IN OUT per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int q; int r; int s; int t;
procedure p2([IN OUT x copia] int t)
int q;
BEGIN
q=s*1;
t=4;
r=s+3;
if q=7 then s=4 else s=s+1;
write(q,r,s,t);
END

procedure p3([IN OUT x copia] int s)
int q; int r;
BEGIN
q=t+1;
r=t+1;
s=4;
t=2;
p4(q);
write(q,r,s,t);
END

procedure p4([MODE] int s)
int q; int r;
BEGIN
q=2;
r=s*1;
s=2;
t=4;
p2(s);
write(q,r,s,t);
END

BEGIN
q=0;
r=2;
s=4;
t=0;
p3(s);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 10

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int a; int b; int c; int d;
procedure p2([MODE] int d)
int a; int b;
procedure p3([IN OUT x rif] int a)
int c; int b;
BEGIN
if a>9 then c=4 else c=3;
b=a;
a=a*1;
d=a-3;
write(a,b,c,d);
END

procedure p4([IN OUT x copia] int a,[IN OUT x copia] int d)
int c;
BEGIN
c=b*1;
a=b*3;
d=c;
b=d*1;
p3(b);
write(a,b,c,d);
END

BEGIN
a=c-1;
b=2;
d=2;
c=1;
p4(c, b);
write(a,b,c,d);
END

BEGIN
a=3;
b=2;
c=3;
d=2;
p2(c);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 1

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int a; int b; int c; int d;
procedure p2([IN OUT x copia] int d,[IN OUT x copia] int c)
int b;
BEGIN
if d=4 then b=3 else b=c;
d=b-3;
c=c;
a=2;
write(a,b,c,d);
END

procedure p3([IN x copia] int c)
int a;
BEGIN
a=4;
c=b;
b=b;
d=4;
p2(b, c);
write(a,b,c,d);
END

procedure p4([MODE] int c)
int d; int b;
BEGIN
d=3;
if a=6 then b=2 else b=c-4;
if d=5 then c=3 else c=4;
a=3;
p3(c);
write(a,b,c,d);
END

BEGIN
a=3;
b=4;
c=3;
d=4;
p4(b);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 2

- Scoping dinamico, MODE = IN per copia
- Scoping statico, MODE = IN per riferimento

```
program p1
int q; int r; int s; int t;
procedure p2([MODE] int q,[IN OUT x copia] int r)
int s;
procedure p3([IN OUT x rif] int t)
int r; int q;
procedure p4([IN x copia] int r)
int s; int q;
BEGIN
s=3;
q=2;
r=t;
t=s;
write(q,r,s,t);
END

BEGIN
r=s;
q=s*1;
t=t-1;
if q>2 then s=q*1 else s=s*1;
p4(t);
write(q,r,s,t);
END

BEGIN
if r>7 then s=r else s=4;
if t>8 then q=q else q=1;
r=3;
t=r;
p3(r);
write(q,r,s,t);
END

BEGIN
q=1;
r=0;
s=2;
t=0;
p2(r, s);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 3

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int a; int b; int c; int d;
procedure p2([IN OUT x copia] int c,[MODE] int d)
int a;
BEGIN
a=b+4;
c=a;
d=c;
b=b*2;
p3(b);
write(a,b,c,d);
END

procedure p3([IN OUT x rif] int b)
int c;
procedure p4([IN OUT x rif] int a)
int c; int b;
BEGIN
c=a;
if d>7 then b=a+1 else b=d+3;
a=d;
d=1;
write(a,b,c,d);
END

BEGIN
c=2;
b=d;
a=a*2;
d=3;
p4(c);
write(a,b,c,d);
END

BEGIN
a=4;
b=3;
c=2;
d=3;
p2(d, a);
write(a,b,c,d);
END
```

#### Passaggio parametri versione n. 4

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN per copia

```
program p1
int a; int b; int c; int d;
procedure p2([MODE] int c,[IN OUT x copia] int b)
int a;
BEGIN
a=d;
c=4;
b=c;
d=b*4;
p3(b);
write(a,b,c,d);
END

procedure p3([IN OUT x rif] int d)
int c;
BEGIN
if b<20 then c=d+2 else c=a*1;
if b<3 then d=c else d=d+2;
a=3;
b=a;
write(a,b,c,d);
END

procedure p4([IN x copia] int b,[IN OUT x rif] int a)
int d;
BEGIN
d=b;
b=4;
a=2;
c=4;
p2(d, a);
write(a,b,c,d);
END

BEGIN
a=2;
b=3;
c=2;
d=4;
p4(b, c);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 5

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int q; int r; int s; int t;
procedure p2([IN OUT x copia] int s)
int r; int t;
BEGIN
r=1;
t=4;
s=3;
if r<1 then q=q else q=q-2;
write(q,r,s,t);
END

procedure p3([IN OUT x copia] int s)
int r;
BEGIN
r=3;
s=t-1;
t=t;
q=3;
p2(t);
write(q,r,s,t);
END

procedure p4([MODE] int q,[IN OUT x rif] int t)
int s;
BEGIN
s=r*3;
q=t+1;
t=3;
r=3;
p3(r);
write(q,r,s,t);
END

BEGIN
q=1;
r=1;
s=4;
t=1;
p4(r, s);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 6

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN per riferimento

```
program p1
int x; int y; int z; int t;
procedure p2([MODE] int t)
int z;
BEGIN
z=x*4;
t=y+2;
x=y;
y=2;
p3(x);
write(x,y,z,t);
END

procedure p3([IN x copia] int z)
int y; int t;
procedure p4([IN OUT x copia] int t,[IN OUT x rif] int z)
int y;
BEGIN
y=z;
t=4;
if z>6 then z=4 else z=x;
x=z*1;
write(x,y,z,t);
END

BEGIN
y=z;
t=4;
z=x;
x=y;
p4(z, t);
write(x,y,z,t);
END

BEGIN
x=1;
y=3;
z=0;
t=1;
p2(x);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 7

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x copia] int t)
int x;
BEGIN
x=t-4;
t=1;
y=z;
z=1;
p3(x);
write(x,y,z,t);
END

procedure p3([IN OUT x copia] int y)
int t;
BEGIN
t=4;
y=x;
z=3;
x=y*2;
p4(t, y);
write(x,y,z,t);
END

procedure p4([IN x copia] int y,[MODE] int z)
int x;
BEGIN
x=y;
y=z-3;
z=4;
if x<50 then t=y-1 else t=3;
write(x,y,z,t);
END

BEGIN
x=2;
y=4;
z=1;
t=0;
p2(z);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 8

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN per copia

```
program p1
int q; int r; int s; int t;
procedure p2([MODE] int s)
int q; int t;
procedure p3([IN OUT x rif] int q,[IN x copia] int t)
int r;
BEGIN
r=s*4;
q=3;
t=t;
s=2;
p4(q);
write(q,r,s,t);
END

BEGIN
q=3;
t=s-4;
s=t+3;
r=1;
p3(s, t);
write(q,r,s,t);
END

procedure p4([IN OUT x copia] int s)
int r;
BEGIN
r=t*1;
if t<10 then s=2 else s=q;
t=s;
q=r;
write(q,r,s,t);
END

BEGIN
q=3;
r=4;
s=1;
t=4;
p2(r);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 9

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int a; int b; int c; int d;
procedure p2([IN x copia] int d,[MODE] int b)
int c;
procedure p3([IN OUT x rif] int a)
int d;
BEGIN
if c<8 then d=a else d=a;
a=c;
b=d+2;
c=2;
p4(c);
write(a,b,c,d);
END

BEGIN
c=d;
d=c;
b=c+2;
a=4;
p3(a);
write(a,b,c,d);
END

procedure p4([IN x copia] int a)
int d;
BEGIN
d=a-4;
a=3;
if a<9 then c=d else c=d-1;
if d<6 then b=2 else b=d+2;
write(a,b,c,d);
END

BEGIN
a=2;
b=0;
c=2;
d=3;
p2(d, a);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 10

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN OUT per copia

```
program p1
int a; int b; int c; int d;
procedure p2([IN x copia] int b,[MODE] int a)
int d;
procedure p3([IN OUT x rif] int b)
int d;
BEGIN
d=3;
b=3;
c=4;
a=1;
write(a,b,c,d);
END

procedure p4([IN x copia] int c)
int a; int b;
BEGIN
a=3;
b=d*1;
c=1;
d=1;
p3(b);
write(a,b,c,d);
END

BEGIN
d=3;
b=a*3;
a=3;
c=a+4;
p4(b);
write(a,b,c,d);
END

BEGIN
a=0;
b=0;
c=0;
d=2;
p2(b, c);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 11

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN per copia

```
program p1
int q; int r; int s; int t;
procedure p2([IN OUT x copia] int r)
int q; int t;
procedure p3([MODE] int s)
int r; int q;
BEGIN
r=t*2;
q=s+3;
s=2;
if s<8 then t=q-4 else t=3;
p4(t, r);
write(q,r,s,t);
END

procedure p4([IN OUT x copia] int t,[IN OUT x rif] int r)
int s;
BEGIN
s=t+1;
t=q*4;
r=q;
q=s-1;
write(q,r,s,t);
END

BEGIN
q=1;
t=s-3;
r=3;
s=1;
p3(q);
write(q,r,s,t);
END

BEGIN
q=1;
r=2;
s=3;
t=4;
p2(q);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 12

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x copia] int t)
int y; int x;
BEGIN
y=1;
x=t*3;
t=x*2;
z=3;
p4(z);
write(x,y,z,t);
END

procedure p3([MODE] int x)
int z;
BEGIN
if y>3 then z=y*3 else z=t;
if t<50 then y=x else y=3;
if x=6 then t=y+4 else t=4;
write(x,y,z,t);
END

procedure p4([IN OUT x copia] int t)
int x;
BEGIN
if t=0 then x=3 else x=3;
t=4;
z=y;
y=t+3;
p3(x);
write(x,y,z,t);
END

BEGIN
x=0;
y=4;
z=2;
t=4;
p2(z);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 13

- Scoping dinamico, MODE = IN per copia
- Scoping statico, MODE = IN per riferimento

```
program p1
int x; int y; int z; int t;
procedure p2([IN x copia] int x,[IN OUT x copia] int y)
int z;
procedure p3([MODE] int z,[IN OUT x rif] int t)
int y;
procedure p4([IN OUT x copia] int y)
int x;
BEGIN
x=t-1;
y=y;
z=3;
if t>50 then t=2 else t=4;
write(x,y,z,t);
END

BEGIN
y=4;
z=3;
t=3;
x=1;
p4(z);
write(x,y,z,t);
END

BEGIN
if t=6 then z=2 else z=x*4;
x=y;
y=y-1;
t=4;
p3(x, y);
write(x,y,z,t);
END

BEGIN
x=0;
y=4;
z=2;
t=1;
p2(t, y);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 14

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int q; int r; int s; int t;
procedure p2([IN x copia] int t,[IN OUT x copia] int q)
int r;
BEGIN
r=q-1;
t=2;
q=3;
s=3;
p3(t);
write(q,r,s,t);
END

procedure p3([IN OUT x copia] int s)
int t; int q;
BEGIN
t=r;
q=2;
s=1;
if t>50 then r=r else r=t;
p4(t);
write(q,r,s,t);
END

procedure p4([MODE] int q)
int t; int r;
BEGIN
t=q+3;
r=s;
q=t-2;
s=t-1;
write(q,r,s,t);
END

BEGIN
q=3;
r=1;
s=0;
t=2;
p2(r, t);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 15

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int a; int b; int c; int d;
procedure p2([IN OUT x rif] int c,[IN OUT x rif] int d)
int b;
BEGIN
b=a;
c=b+2;
d=1;
a=a;
p4(a);
write(a,b,c,d);
END

procedure p3([MODE] int a,[IN OUT x rif] int d)
int c;
BEGIN
c=4;
a=2;
d=2;
b=4;
write(a,b,c,d);
END

procedure p4([IN x copia] int c)
int a;
BEGIN
a=4;
c=c;
d=c;
b=3;
p3(d, b);
write(a,b,c,d);
END

BEGIN
a=0;
b=2;
c=3;
d=1;
p2(d, c);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 16

- Scoping dinamico, MODE = IN OUT per copia
- Scoping statico, MODE = IN per riferimento

```
program p1
int a; int b; int c; int d;
procedure p2([MODE] int b)
int c;
procedure p3([IN OUT x rif] int b,[IN x copia] int d)
int c;
procedure p4([IN OUT x rif] int d)
int a;
BEGIN
a=3;
d=1;
c=a*3;
if b<9 then b=b*1 else b=d*2;
write(a,b,c,d);
END

BEGIN
c=1;
b=2;
d=c;
a=4;
p4(c);
write(a,b,c,d);
END

BEGIN
c=d+1;
b=2;
a=4;
d=b;
p3(c, a);
write(a,b,c,d);
END

BEGIN
a=2;
b=4;
c=2;
d=2;
p2(d);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 17

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int a; int b; int c; int d;
procedure p2([IN OUT x rif] int c,[IN OUT x rif] int d)
int b;
procedure p3([MODE] int c,[IN OUT x copia] int b)
int a;
BEGIN
a=2;
c=b-4;
b=a;
d=b-1;
p4(a, d);
write(a,b,c,d);
END

BEGIN
b=a+1;
c=2;
d=c;
a=b;
p3(a, d);
write(a,b,c,d);
END

procedure p4([IN OUT x copia] int d,[IN x copia] int c)
int a;
BEGIN
a=2;
if d<7 then d=a else d=1;
c=1;
b=c*4;
write(a,b,c,d);
END

BEGIN
a=4;
b=4;
c=4;
d=4;
p2(c, a);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 18

- Scoping dinamico, MODE = IN OUT per copia
- Scoping statico, MODE = OUT per copia

```
program p1
int q; int r; int s; int t;
procedure p2([MODE] int r,[IN OUT x rif] int t)
int s;
BEGIN
s=r;
r=2;
t=2;
q=1;
write(q,r,s,t);
END

procedure p3([IN x copia] int t)
int s; int r;
procedure p4([IN x copia] int r)
int t;
BEGIN
t=s+1;
r=r;
if s=6 then q=2 else q=3;
if q<1 then s=r*1 else s=q;
p2(s, q);
write(q,r,s,t);
END

BEGIN
s=2;
r=t;
t=4;
q=r+1;
p4(q);
write(q,r,s,t);
END

BEGIN
q=0;
r=1;
s=1;
t=2;
p3(s);
write(q,r,s,t);
END
```

**Versione 1 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 2 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

**Versione 3 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 4 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 5 dell'esercizio su programmazione ML**

Scrivere in ML la funzione map.

**Versione 6 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 7 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

**Versione 8 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 9 dell'esercizio su programmazione ML**

Scrivere in ML la funzione map.

**Versione 10 dell'esercizio su programmazione ML**

Scrivere in ML la funzione map.

**Versione 1 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 2 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

**Versione 3 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 4 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

**Versione 5 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 6 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

**Versione 7 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 8 dell'esercizio su programmazione ML**

Scrivere in ML la funzione map.

**Versione 9 dell'esercizio su programmazione ML**

Scrivere in ML la funzione map.

**Versione 10 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

**Versione 11 dell'esercizio su programmazione ML**

Scrivere in ML la funzione map.

**Versione 1 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

**Versione 2 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

**Versione 3 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 4 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 5 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 6 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

**Versione 7 dell'esercizio su programmazione ML**

Scrivere in ML la funzione map.

**Versione 8 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 9 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 10 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 11 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

**Versione 12 dell'esercizio su programmazione ML**

Scrivere in ML la funzione map.

**Versione 13 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 14 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 15 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 16 dell'esercizio su programmazione ML**

Scrivere in ML la funzione map.

**Versione 17 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 18 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

### **Versione 1 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lunghezza di una lista.

### **Versione 2 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola il minimo degli elementi di una lista.

### **Versione 3 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che divide tutti gli elementi di una lista di interi per la lunghezza della lista.

#### **Versione 4 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola il massimo degli elementi di una lista.

### **Versione 5 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lunghezza di una lista.

### **Versione 6 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che divide tutti gli elementi di una lista di interi per la lunghezza della lista.

### **Versione 7 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi pari di una lista.

### **Versione 8 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi positivi di una lista.

### **Versione 9 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola il prodotto degli elementi di una lista.

### **Versione 10 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi negativi di una lista.

### **Versione 1 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola il massimo degli elementi di una lista.

### **Versione 2 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che incrementa tutti gli elementi di una lista di interi.

### **Versione 3 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi non negativi di una lista.

#### **Versione 4 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi divisibili per 3 di una lista.

### **Versione 5 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi pari di una lista.

### **Versione 6 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi pari di una lista.

### **Versione 7 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la somma degli elementi di una lista.

### **Versione 8 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la somma degli elementi di una lista.

### **Versione 9 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi non negativi di una lista.

### **Versione 10 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi pari di una lista.

### **Versione 11 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la concatenazione di due liste.

### **Versione 1 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi dispari di una lista.

### **Versione 2 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che divide tutti gli elementi di una lista di interi per la lunghezza della lista.

### **Versione 3 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola il massimo degli elementi di una lista.

#### **Versione 4 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la somma degli elementi di una lista.

### **Versione 5 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la media degli elementi di una lista.

### **Versione 6 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la media degli elementi di una lista.

### **Versione 7 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la concatenazione di due liste.

### **Versione 8 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che decrementa tutti gli elementi di una lista di interi.

### **Versione 9 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi pari di una lista.

### **Versione 10 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi non negativi di una lista.

### **Versione 11 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi non negativi di una lista.

**Versione 12 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola l'inversa di una lista.

### **Versione 13 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la concatenazione di due liste.

### **Versione 14 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che decrementa tutti gli elementi di una lista di interi.

### **Versione 15 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che divide tutti gli elementi di una lista di interi per la lunghezza della lista.

### **Versione 16 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che divide tutti gli elementi di una lista di interi per la lunghezza della lista.

### **Versione 17 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi dispari di una lista.

### **Versione 18 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi negativi di una lista.

### **Versione 1 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Nell'overloading due metodi possono avere lo stesso nome e lo stesso numero di parametri
- B) Un array ha una sola superclasse
- C) Il minimo numero di elementi che un array può contenere è 0
- D) Una classe non può essere dichiarata `abstract` o `final`
- E) Una classe può essere dichiarata `private`

## **Versione 2 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) È possibile dichiarare un attributo senza inizializzarlo
- B) Due metodi possono avere diverso nome ma lo stesso tipo di parametri
- C) Una classe può essere dichiarata final
- D) Un attributo static non può essere acceduto mediante un riferimento a un oggetto della sua classe di appartenenza
- E) La lunghezza di un array non può essere variata dopo la sua creazione

### **Versione 3 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Un oggetto esiste sempre dopo la sua dichiarazione
- B) Tutti i tipi di eccezioni estendono la classe `Throwable`
- C) Un array ha un'unica superclasse
- D) Una classe può essere dichiarata `protected`
- E) Un attributo `static` può essere acceduto mediante un riferimento a un oggetto della sua classe di appartenenza

#### **Versione 4 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) La dimensione di un array può non essere indicata al momento della dichiarazione dell'array
- B) Un attributo `static` non può essere acceduto mediante il nome della sua classe di appartenenza
- C) Un array vuoto non può avere riferimento `null`
- D) La dichiarazione di un oggetto e la sua creazione possono essere svolte contemporaneamente
- E) Non tutti i tipi di eccezioni estendono la classe `RuntimeException`

### **Versione 5 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) La lunghezza di un array non può essere variata dopo la sua costruzione
- B) Un array con riferimento `null` non ha lunghezza
- C) Due metodi non possono differire per il tipo di ritorno
- D) Una classe può essere dichiarata `private`
- E) La lunghezza di un array non può essere variata dopo la sua creazione

### **Versione 6 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Nell'overloading due metodi possono avere lo stesso nome ma diverso tipo di parametri
- B) Un attributo `static` può essere acceduto mediante il nome della sua classe di appartenenza
- C) Un array ha più di una superclasse
- D) Un array ha un'unica superclasse
- E) Ogni valore in memoria è associato ad un tipo di dato particolare

### **Versione 7 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) La dichiarazione di un oggetto e la sua creazione possono essere svolte solo in tempi diversi
- B) Due metodi non possono avere diverso nome ma lo stesso tipo di parametri
- C) Un array ha più di una superclasse
- D) Tutti i tipi di eccezioni estendono la classe `RuntimeException`
- E) Prima dell'esecuzione del costruttore alle variabili di istanza sono assegnati i valori di default

### **Versione 8 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Una classe non può essere dichiarata **final**
- B) Una classe non interna può essere dichiarata **private**
- C) Un array vuoto può avere riferimento **null**
- D) Un attributo non può essere contemporaneamente **static** e **final**
- E) Una classe può essere dichiarata **protected**

### **Versione 9 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Un array con riferimento `null` non ha lunghezza
- B) Tutti i tipi di eccezioni estendono la classe `RuntimeException`
- C) I modificatori applicati a una variabile di tipo array si applicano alla variabile array e non ai suoi elementi
- D) Due metodi possono avere diverso nome ma lo stesso tipo di parametri
- E) La dimensione di un array può non essere indicata al momento della dichiarazione dell'array

### **Versione 10 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Non tutti i tipi di eccezioni estendono la classe `RuntimeException`
- B) Un attributo non può essere contemporaneamente `static` e `private`
- C) Una classe non può essere dichiarata `abstract` o `final`
- D) Un attributo `static` non può essere acceduto mediante un riferimento a un oggetto della sua classe di appartenenza
- E) Il minimo numero di elementi che un array può contenere è 1

### **Versione 11 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Non tutti i tipi di eccezioni estendono la classe `RuntimeException`
- B) Tutti i tipi di eccezioni estendono la classe `Object`
- C) Un array vuoto può avere riferimento `null`
- D) Un attributo può essere contemporaneamente `static` e `final`
- E) Un attributo `static` può essere acceduto mediante il nome della sua classe di appartenenza

### **Versione 12 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Un oggetto esiste sempre dopo la sua dichiarazione
- B) Subito dopo l'esecuzione del costruttore alle variabili di istanza sono assegnati i valori di default
- C) La dichiarazione di un oggetto e la sua creazione possono essere svolte solo in tempi diversi
- D) La lunghezza di un array non può essere variata dopo la sua costruzione
- E) La dichiarazione di un oggetto e la sua creazione possono essere svolte solo contemporaneamente

### **Versione 13 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Due metodi possono differire per il tipo di ritorno
- B) La lunghezza di un array non può essere variata dopo la sua creazione
- C) La lunghezza di un array non può essere variata dopo la sua costruzione
- D) Non tutti i tipi di eccezioni estendono la classe `Object`
- E) Una classe può essere dichiarata `protected`

### **Versione 14 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Una classe non interna può essere dichiarata `private`
- B) L'operatore `new` non restituisce un riferimento all'oggetto appena creato
- C) I modificatori applicati a una variabile di tipo array non si applicano alla variabile array ma ai suoi elementi
- D) Un array non possiede dei membri
- E) Un array ha un'unica superclasse

### **Versione 15 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Quando due metodi hanno lo stesso nome si ha overloading
- B) Una classe non può essere dichiarata **abstract** o **final**
- C) La dimensione di un array deve essere indicata al momento della dichiarazione dell'array
- D) Un attributo non può essere contemporaneamente **static** e **private**
- E) Un attributo **static** può essere acceduto mediante un riferimento a un oggetto della sua classe di appartenenza

### **Versione 16 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) In Java non tutti i tipi numerici sono con segno
- B) Il minimo numero di elementi che un array può contenere è 1
- C) Due metodi non possono differire per il tipo di ritorno
- D) Tutti i tipi di eccezioni estendono la classe Object
- E) Una classe non può essere dichiarata `abstract` o `final`

### **Versione 17 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Ogni valore in memoria non è associato ad un tipo di dato particolare
- B) Un attributo non può essere contemporaneamente **static** e **private**
- C) Un array ha più di una superclasse
- D) Subito dopo l'esecuzione del costruttore alle variabili di istanza sono assegnati i valori di default
- E) Un attributo **static** può essere acceduto mediante il nome della sua classe di appartenenza

### **Versione 18 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) La lunghezza di un array può essere variata dopo la sua costruzione
- B) Non tutti i tipi di eccezioni estendono la classe `Throwable`
- C) Nell'overloading due metodi non possono avere lo stesso nome ma diverso tipo di parametri
- D) I modificatori applicati a una variabile di tipo array si applicano alla variabile array e non ai suoi elementi
- E) Un array vuoto può avere riferimento `null`

### **Versione 19 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Una classe può essere dichiarata **private**
- B) Un attributo **static** può essere acceduto mediante un riferimento a un oggetto della sua classe di appartenenza
- C) Tutti i tipi di eccezioni estendono la classe **Object**
- D) Non è possibile dichiarare un array senza indicarne la dimensione
- E) Due metodi possono differire per il tipo di ritorno

### **Versione 20 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Non è possibile dichiarare un attributo senza inizializzarlo
- B) L'operatore `new` non restituisce un riferimento all'oggetto appena creato
- C) Nell'overloading due metodi non possono avere lo stesso nome ma diverso tipo di parametri
- D) Ai metodi `static` non si applica il *dynamic method dispatch*
- E) Una classe non può essere dichiarata `protected`

### **Versione 21 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) I modificatori applicati a una variabile di tipo array si applicano alla variabile array e non ai suoi elementi
- B) I modificatori che precedono una variabile di tipo array si applicano alla variabile array e non ai suoi elementi
- C) La dimensione di un array può non essere indicata al momento della dichiarazione dell'array
- D) Non tutti i tipi di eccezioni estendono la classe `Object`
- E) La lunghezza di un array non può essere variata dopo la sua creazione

### **Versione 22 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Il minimo numero di elementi che un array può contenere è 1
- B) Un array con riferimento `null` non ha lunghezza
- C) Un array ha una sola superclasse
- D) Un array ha un'unica superclasse
- E) È possibile dichiarare un array senza indicarne la dimensione

### **Versione 23 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) La dimensione di un array può non essere indicata al momento della dichiarazione dell'array
- B) Un attributo può essere contemporaneamente **static** e **final**
- C) Un attributo può essere contemporaneamente **static** e **private**
- D) Una classe non può essere dichiarata **protected**
- E) Un array vuoto non può avere riferimento **null**

### **Versione 24 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Non tutti i tipi di eccezioni estendono la classe `Object`
- B) L'operatore `new` non restituisce un riferimento all'oggetto appena creato
- C) Nell'overloading due metodi possono avere lo stesso nome e lo stesso numero di parametri
- D) La lunghezza di un array può essere variata dopo la sua costruzione
- E) Due metodi non possono avere diverso nome ma lo stesso tipo di parametri

### **Versione 25 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) La lunghezza di un array non può essere variata dopo la sua costruzione
- B) Un oggetto può non esistere dopo la sua dichiarazione
- C) Non tutti i tipi di eccezioni estendono la classe `RuntimeException`
- D) Prima dell'esecuzione del costruttore alle variabili di istanza sono assegnati i valori di default
- E) Quando due metodi hanno lo stesso nome si ha overloading

### **Versione 26 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) L'operatore `new` non restituisce un riferimento all'oggetto appena creato
- B) Una classe non può essere dichiarata `protected`
- C) La dichiarazione di un oggetto e la sua creazione possono essere svolte solo in tempi diversi
- D) Un array ha un'unica superclasse
- E) Non è possibile dichiarare un array senza indicarne la dimensione

### **Versione 27 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) La lunghezza di un array può essere variata dopo la sua costruzione
- B) In Java non tutti i tipi numerici sono con segno
- C) Una classe non può essere dichiarata **final**
- D) Nell'overloading due metodi possono avere lo stesso nome e lo stesso numero di parametri
- E) Un array vuoto può avere riferimento **null**

### **Versione 28 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Una classe non può essere dichiarata `final`
- B) La dimensione di un array può non essere indicata al momento della dichiarazione dell'array
- C) Ogni valore in memoria non è associato ad un tipo di dato particolare
- D) Non tutti i tipi di eccezioni estendono la classe `Throwable`
- E) Un array non possiede dei membri

### **Versione 29 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Tutti i tipi di eccezioni estendono la classe `Throwable`
- B) Nell'overloading due metodi non possono avere lo stesso nome ma diverso tipo di parametri
- C) Una classe non può essere dichiarata `protected`
- D) Una classe non interna può essere dichiarata `private`
- E) Un attributo non può essere contemporaneamente `static` e `private`

### **Versione 1 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Un attributo non può essere contemporaneamente `static` e `final`
- B) La lunghezza di un array non può essere variata dopo la sua costruzione
- C) L'operatore `new` restituisce un riferimento all'oggetto appena creato
- D) Un array vuoto non può avere riferimento `null`
- E) Prima dell'esecuzione del costruttore alle variabili di istanza sono assegnati i valori di default

## **Versione 2 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Una classe può essere dichiarata `final`
- B) I modificatori applicati a una variabile di tipo array si applicano alla variabile array e non ai suoi elementi
- C) Un attributo `static` non può essere acceduto mediante il nome della sua classe di appartenenza
- D) Un array vuoto non può avere riferimento `null`
- E) I modificatori che precedono una variabile di tipo array si applicano alla variabile array e non ai suoi elementi

### **Versione 3 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) I modificatori applicati a una variabile di tipo array si applicano alla variabile array e non ai suoi elementi
- B) I modificatori che precedono una variabile di tipo array si applicano alla variabile array e non ai suoi elementi
- C) Un array possiede dei membri
- D) Il minimo numero di elementi che un array può contenere è 1
- E) È possibile dichiarare un array senza indicarne la dimensione

#### **Versione 4 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Nell'overloading due metodi possono avere lo stesso nome ma diverso tipo di parametri
- B) Una classe non può essere dichiarata `protected`
- C) Una classe non può essere dichiarata `abstract` o `final`
- D) Non tutti i tipi di eccezioni estendono la classe `Object`
- E) Ogni valore in memoria non è associato ad un tipo di dato particolare

### **Versione 5 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) I modificatori che precedono una variabile di tipo array si applicano alla variabile array ed anche ai suoi elementi
- B) Non è possibile dichiarare un array senza indicarne la dimensione
- C) Due metodi possono avere diverso nome ma lo stesso tipo di parametri
- D) Una classe non può essere dichiarata **private**
- E) Il minimo numero di elementi che un array può contenere è 1

### **Versione 6 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Un array non possiede dei membri
- B) Un attributo **static** non può essere acceduto mediante il nome della sua classe di appartenenza
- C) Una classe non può essere dichiarata **protected**
- D) La lunghezza di un array può essere variata dopo la sua creazione
- E) Tutti i tipi di eccezioni estendono la classe **Throwable**

### **Versione 7 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) I modificatori applicati a una variabile di tipo array non si applicano alla variabile array ma ai suoi elementi
- B) Un attributo **static** non può essere acceduto mediante il nome della sua classe di appartenenza
- C) In Java tutti i tipi numerici sono con segno
- D) Un attributo **static** non può essere acceduto mediante un riferimento a un oggetto della sua classe di appartenenza
- E) Una classe non può essere dichiarata **abstract** o **final**

### **Versione 8 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Un array non ha un'unica superclasse
- B) Un attributo **static** non può essere acceduto mediante il nome della sua classe di appartenenza
- C) Un oggetto può non esistere dopo la sua dichiarazione
- D) Non è possibile dichiarare un array senza indicarne la dimensione
- E) Un array con riferimento **null** ha lunghezza zero

### **Versione 9 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) In Java non tutti i tipi numerici sono con segno
- B) Un array vuoto può avere riferimento `null`
- C) Un array non possiede dei membri
- D) Una classe non può essere dichiarata `abstract` o `final`
- E) Il minimo numero di elementi che un array può contenere è 0

### **Versione 10 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Un array ha più di una superclasse
- B) Non è possibile dichiarare un attributo senza inizializzarlo
- C) Una classe non può essere dichiarata `protected`
- D) La lunghezza di un array può essere variata dopo la sua costruzione
- E) Un array vuoto non può avere riferimento `null`

### **Versione 11 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Un array con riferimento `null` non ha lunghezza
- B) Un attributo può essere contemporaneamente `static` e `private`
- C) Subito dopo l'esecuzione del costruttore alle variabili di istanza sono assegnati i valori di default
- D) Un array ha una sola superclasse
- E) Un attributo `static` può essere acceduto mediante il nome della sua classe di appartenenza

### **Versione 12 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Nell'overloading due metodi possono avere lo stesso nome ma diverso tipo di parametri
- B) Una classe non può essere dichiarata `abstract` o `final`
- C) È possibile dichiarare un attributo senza inizializzarlo
- D) La dimensione di un array può non essere indicata al momento della dichiarazione dell'array
- E) Un array vuoto non può avere riferimento `null`

### **Versione 13 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Ai metodi **static** si applica il *dynamic method dispatch*
- B) Due metodi non possono differire per il tipo di ritorno
- C) Un oggetto esiste sempre dopo la sua dichiarazione
- D) Un array ha una sola superclasse
- E) Un array non possiede dei membri

### **Versione 14 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) Nell'overloading due metodi non possono avere lo stesso nome ma diverso tipo di parametri
- B) Un array non ha un'unica superclasse
- C) Un array non possiede dei membri
- D) Due metodi possono differire per il tipo di ritorno
- E) Un attributo non può essere contemporaneamente **static** e **private**

### **Versione 15 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) Ogni valore in memoria non è associato ad un tipo di dato particolare
- B) Quando due metodi hanno lo stesso nome non sempre si ha overloading
- C) Non tutti i tipi di eccezioni estendono la classe `RuntimeException`
- D) Prima dell'esecuzione del costruttore alle variabili di istanza sono assegnati i valori di default
- E) Un attributo `static` può essere acceduto mediante un riferimento a un oggetto della sua classe di appartenenza

### **Versione 16 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) La dimensione di un array può non essere indicata al momento della dichiarazione dell'array
- B) Nell'overloading due metodi non possono avere lo stesso nome ma diverso tipo di parametri
- C) Una classe non può essere dichiarata **protected**
- D) Il minimo numero di elementi che un array può contenere è 1
- E) Quando due metodi hanno lo stesso nome si ha overloading

### **Versione 17 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) In Java tutti i tipi numerici sono con segno
- B) La lunghezza di un array non può essere variata dopo la sua costruzione
- C) Un attributo non può essere contemporaneamente `static` e `private`
- D) La dichiarazione di un oggetto e la sua creazione possono essere svolte contemporaneamente
- E) Prima dell'esecuzione del costruttore alle variabili di istanza sono assegnati i valori di default

### **Versione 18 dell'esercizio 1**

Dire quale delle seguenti affermazioni è falsa:

- A) La dichiarazione di un oggetto e la sua creazione possono essere svolte solo contemporaneamente
- B) L'operatore `new` restituisce un riferimento all'oggetto appena creato
- C) Un oggetto può non esistere dopo la sua dichiarazione
- D) Nell'overloading due metodi possono avere lo stesso nome e lo stesso numero di parametri
- E) Una classe può essere dichiarata `abstract` o `final`

### **Versione 19 dell'esercizio 1**

Dire quale delle seguenti affermazioni è vera:

- A) La lunghezza di un array può essere variata dopo la sua costruzione
- B) La lunghezza di un array può essere variata dopo la sua creazione
- C) Un array vuoto può avere riferimento null
- D) In Java non tutti i tipi numerici sono con segno
- E) Nell'overloading due metodi possono avere lo stesso nome ma diverso tipo di parametri

### **Versione 1 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] a;  
Integer [] g;  
Exception [] r;  
g = new Integer [7];  
a = new Exception [1];  
r = new Exception [6];
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) r = (Exception []) a;
- B) r = (Exception []) g;
- C) g = (Integer []) a;
- D) g = (Integer []) r;
- E) Nessuno dei precedenti

## **Versione 2 dell'esercizio 2**

Date le dichiarazioni:

```
Object c;  
Boolean x;  
Error y;  
c = new Error();  
y = new Error();  
x = new Boolean(true);
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) x = (Boolean) y;
- B) y = (Error) x;
- C) y = (Error) c;
- D) x = (Boolean) c;
- E) Nessuno dei precedenti

### **Versione 3 dell'esercizio 2**

Date le dichiarazioni:

```
Exception [] g;  
Boolean [] u;  
Object [] x;  
x = new Boolean [5];  
g = new Exception [2];  
u = new Boolean [9];
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) g = (Exception []) u;
- B) g = (Exception []) x;
- C) u = (Boolean []) x;
- D) u = (Boolean []) g;
- E) Nessuno dei precedenti

### **Versione 4 dell'esercizio 2**

Date le dichiarazioni:

```
Boolean c;  
Integer d;  
Object m;  
m = new Boolean(false);  
d = new Integer(20);  
c = new Boolean(true);
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) `c = (Boolean) d;`
- B) `c = (Boolean) m;`
- C) `d = (Integer) c;`
- D) `d = (Integer) m;`
- E) Nessuno dei precedenti

### **Versione 5 dell'esercizio 2**

Date le dichiarazioni:

```
Error p;  
Exception w;  
Object z;  
p = new Error();  
w = new Exception();  
z = new Error();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) w = (Exception) z;
- B) p = (Error) z;
- C) p = (Error) w;
- D) w = (Exception) p;
- E) Nessuno dei precedenti

## Versione 6 dell'esercizio 2

Date le dichiarazioni:

```
class Sub2 extends C0 {...}
class B2 extends C0 {...}
class C0 extends Object {...}
```

e le inizializzazioni di variabile:

```
Sub2 a;
C0 b;
B2 r;
b = new B2();
r = new B2();
a = new Sub2();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) a = (Sub2) b;
- B) a = (Sub2) r;
- C) r = (B2) a;
- D) r = (B2) b;
- E) Nessuno dei precedenti

### **Versione 7 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] [] e;
Integer [] v;
Object [] z;
e = new Object [1] [3];
z = new Object [2] [4];
v = new Integer [6];
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) `e = (Object []) [] z;`
- B) `e = (Object []) [] v;`
- C) `v = (Integer []) e;`
- D) `v = (Integer []) z;`
- E) Nessuno dei precedenti

### **Versione 8 dell'esercizio 2**

Date le dichiarazioni:

```
Integer [] f;  
Object [] t;  
Object [] [] u;  
u = new Object [2] [0];  
f = new Integer [3];  
t = new Object [6] [5];
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) `f = (Integer []) u;`
- B) `u = (Object []) [] t;`
- C) `u = (Object []) [] f;`
- D) `f = (Integer []) t;`
- E) Nessuno dei precedenti

### **Versione 9 dell'esercizio 2**

Date le dichiarazioni:

```
class C0 extends Object {...}
class B2 extends C0 {...}
class Sub1 extends C0 {...}
```

e le inizializzazioni di variabile:

```
B2 m;
Sub1 r;
C0 u;
m = new B2();
u = new B2();
r = new Sub1();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) m = (B2) r;
- B) r = (Sub1) m;
- C) m = (B2) u;
- D) r = (Sub1) u;
- E) Nessuno dei precedenti

### **Versione 10 dell'esercizio 2**

Date le dichiarazioni:

```
Exception [] d;  
Object [] [] r;  
Object [] t;  
t = new Object [2] [7];  
d = new Exception [9];  
r = new Object [0] [2];
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) `d = (Exception []) t;`
- B) `d = (Exception []) r;`
- C) `r = (Object [] []) t;`
- D) `r = (Object [] []) d;`
- E) Nessuno dei precedenti

### **Versione 11 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] d;  
String [] h;  
Object [] [] m;  
d = new Object [3] [5];  
m = new Object [4] [5];  
h = new String [7];
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) m = (Object []) [] h;
- B) h = (String []) m;
- C) h = (String []) d;
- D) m = (Object []) [] d;
- E) Nessuno dei precedenti

### **Versione 12 dell'esercizio 2**

Date le dichiarazioni:

```
class B2 extends Super {...}
class Super extends Object {...}
class Sub1 extends Super {...}
```

e le inizializzazioni di variabile:

```
B2 s;
Sub1 x;
Super y;
x = new Sub1();
s = new B2();
y = new Sub1();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) x = (Sub1) s;
- B) s = (B2) x;
- C) x = (Sub1) y;
- D) s = (B2) y;
- E) Nessuno dei precedenti

### **Versione 13 dell'esercizio 2**

Date le dichiarazioni:

```
String n;  
Boolean p;  
Object t;  
p = new Boolean(false);  
n = new String("abcd");  
t = new Boolean(true);
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) p = (Boolean) t;
- B) p = (Boolean) n;
- C) n = (String) p;
- D) n = (String) t;
- E) Nessuno dei precedenti

### **Versione 14 dell'esercizio 2**

Date le dichiarazioni:

```
Boolean [] e;  
Integer [] v;  
Object [] x;  
x = new Integer [6];  
e = new Boolean [0];  
v = new Integer [2];
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) `e = (Boolean []) x;`
- B) `v = (Integer []) e;`
- C) `v = (Integer []) x;`
- D) `e = (Boolean []) v;`
- E) Nessuno dei precedenti

## Versione 15 dell'esercizio 2

Date le dichiarazioni:

```
class A extends Object {...}
class B1 extends A {...}
class B2 extends A {...}
```

e le inizializzazioni di variabile:

```
A c;
B1 e;
B2 m;
m = new B2();
c = new B2();
e = new B1();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) e = (B1) m;
- B) e = (B1) c;
- C) m = (B2) e;
- D) m = (B2) c;
- E) Nessuno dei precedenti

## Versione 16 dell'esercizio 2

Date le dichiarazioni:

```
class B2 extends A {...}
class A extends Object {...}
class B1 extends A {...}
```

e le inizializzazioni di variabile:

```
B2 d;
B1 f;
A z;
z = new B1();
d = new B2();
f = new B1();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) f = (B1) d;
- B) d = (B2) f;
- C) d = (B2) z;
- D) f = (B1) z;
- E) Nessuno dei precedenti

## Versione 17 dell'esercizio 2

Date le dichiarazioni:

```
Object [] g;  
Object [] [] w;  
Exception [] x;  
g = new Object [1] [6];  
x = new Exception [4];  
w = new Object [7] [9];
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) x = (Exception []) g;
- B) x = (Exception []) w;
- C) w = (Object [] []) x;
- D) w = (Object [] []) g;
- E) Nessuno dei precedenti

### **Versione 18 dell'esercizio 2**

Date le dichiarazioni:

```
String g;
Object s;
Boolean t;
t = new Boolean(true);
s = new String("abcde");
g = new String("");
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) t = (Boolean) s;
- B) t = (Boolean) g;
- C) g = (String) t;
- D) g = (String) s;
- E) Nessuno dei precedenti

### **Versione 19 dell'esercizio 2**

Date le dichiarazioni:

```
class A extends Object {...}
class B1 extends A {...}
class B extends A {...}
```

e le inizializzazioni di variabile:

```
B1 a;
B b;
A t;
b = new B();
a = new B1();
t = new B1();
```

indicare quale dei seguenti assegnamenti è corretto a tempo di esecuzione.

- A) a = (B1) b;
- B) a = (B1) t;
- C) b = (B) t;
- D) b = (B) a;
- E) Nessuno dei precedenti

### **Versione 1 dell'esercizio 2**

Date le dichiarazioni:

```
Exception [] h;  
Object [] r;  
Integer [] w;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `h = w;`
- B) `h = (Exception []) w;`
- C) `w = r;`
- D) `r = (Object []) w;`
- E) `h = r;`

## **Versione 2 dell'esercizio 2**

Date le dichiarazioni:

```
Exception [] p;  
Integer [] r;  
Object [] t;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `p = t;`
- B) `r = (Integer []) p;`
- C) `r = p;`
- D) `t = (Object []) p;`
- E) `r = t;`

### **Versione 3 dell'esercizio 2**

Date le dichiarazioni:

```
Object a;  
Error p;  
String x;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) p = x;
- B) p = a;
- C) p = (Error) x;
- D) a = (Object) x;
- E) x = a;

#### **Versione 4 dell'esercizio 2**

Date le dichiarazioni:

```
Error r;  
Boolean y;  
Object z;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `z = y;`
- B) `r = y;`
- C) `r = z;`
- D) `r = (Error) y;`
- E) `y = (Boolean) r;`

### **Versione 5 dell'esercizio 2**

Date le dichiarazioni:

```
class C0 extends Object {...}
class C1 extends C0 {...}
class Sub1 extends C0 {...}
```

e le dichiarazioni di variabile:

```
C1 r;
C0 s;
Sub1 z;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `r = s;`
- B) `r = (C1) z;`
- C) `s = (C0) z;`
- D) `r = z;`
- E) `z = (Sub1) r;`

### **Versione 6 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] [] c;  
Integer [] g;  
Object [] w;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `g = w;`
- B) `w = (Object []) g;`
- C) `c = g;`
- D) `g = (Integer []) c;`
- E) `g = c;`

### **Versione 7 dell'esercizio 2**

Date le dichiarazioni:

```
Object b;  
Error f;  
String p;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `f = p;`
- B) `p = (String) f;`
- C) `b = f;`
- D) `f = b;`
- E) `p = b;`

### **Versione 8 dell'esercizio 2**

Date le dichiarazioni:

```
Integer e;  
Error n;  
Object x;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) e = (Integer) n;
- B) n = (Error) e;
- C) e = n;
- D) n = x;
- E) e = (Integer) x;

### **Versione 9 dell'esercizio 2**

Date le dichiarazioni:

```
Exception [] b;  
Object [] c;  
String [] v;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `b = c;`
- B) `v = (String []) c;`
- C) `v = (String []) b;`
- D) `v = b;`
- E) `b = v;`

### **Versione 10 dell'esercizio 2**

Date le dichiarazioni:

```
class A extends Object {...}
class Sub1 extends A {...}
class B extends A {...}
```

e le dichiarazioni di variabile:

```
B e;
Sub1 w;
A z;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) e = (B) w;
- B) e = z;
- C) w = (Sub1) z;
- D) w = (Sub1) e;
- E) e = w;

### **Versione 11 dell'esercizio 2**

Date le dichiarazioni:

```
Integer g;  
Object h;  
Boolean z;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `g = (Integer) z;`
- B) `h = g;`
- C) `z = h;`
- D) `g = h;`
- E) `z = g;`

### **Versione 12 dell'esercizio 2**

Date le dichiarazioni:

```
Object c;  
String n;  
Error z;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `z = n;`
- B) `n = z;`
- C) `n = (String) z;`
- D) `c = z;`
- E) `z = c;`

### **Versione 13 dell'esercizio 2**

Date le dichiarazioni:

```
Error [] q;  
Integer [] t;  
Object [] x;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `q = x;`
- B) `t = q;`
- C) `t = (Integer []) q;`
- D) `t = x;`
- E) `t = (Integer []) x;`

### **Versione 14 dell'esercizio 2**

Date le dichiarazioni:

```
Error e;  
String q;  
Object y;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) y = (Object) e;
- B) e = q;
- C) q = y;
- D) q = e;
- E) e = y;

### **Versione 15 dell'esercizio 2**

Date le dichiarazioni:

```
class C0 extends Object {...}
class B1 extends C0 {...}
class Sub2 extends C0 {...}
```

e le dichiarazioni di variabile:

```
C0 h;
B1 m;
Sub2 t;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `t = m;`
- B) `t = h;`
- C) `m = t;`
- D) `m = (B1) h;`
- E) `m = h;`

### **Versione 16 dell'esercizio 2**

Date le dichiarazioni:

```
Error d;  
Boolean q;  
Object u;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) d = (Error) q;
- B) q = u;
- C) q = (Boolean) d;
- D) q = d;
- E) u = (Object) d;

### **Versione 17 dell'esercizio 2**

Date le dichiarazioni:

```
String [] d;  
Object [] u;  
Error [] v;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) d = (String []) v;
- B) v = (Error []) d;
- C) d = u;
- D) v = (Error []) u;
- E) v = d;

### **Versione 18 dell'esercizio 2**

Date le dichiarazioni:

```
Exception [] a;  
Object [] c;  
Error [] x;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `x = (Error []) a;`
- B) `c = (Object []) x;`
- C) `a = x;`
- D) `a = (Exception []) x;`
- E) `a = c;`

### **Versione 19 dell'esercizio 2**

Date le dichiarazioni:

```
Object r;  
Integer v;  
String w;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `v = r;`
- B) `w = (String) v;`
- C) `w = (String) r;`
- D) `v = w;`
- E) `w = v;`

### **Versione 20 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] a;  
Integer [] d;  
Object [][] v;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `d = v;`
- B) `d = a;`
- C) `d = (Integer []) v;`
- D) `v = (Object []) d;`
- E) `a = d;`

### **Versione 21 dell'esercizio 2**

Date le dichiarazioni:

```
class C0 extends Object {...}
class C1 extends C0 {...}
class B1 extends C0 {...}
```

e le dichiarazioni di variabile:

```
C0 a;
B1 e;
C1 r;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `e = r;`
- B) `r = a;`
- C) `r = (C1) e;`
- D) `e = (B1) a;`
- E) `e = (B1) r;`

## **Versione 22 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] f;  
Error [] n;  
Boolean [] q;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) n = (Error []) q;
- B) q = (Boolean []) n;
- C) q = f;
- D) f = (Object []) q;
- E) n = q;

### **Versione 23 dell'esercizio 2**

Date le dichiarazioni:

```
Error [] e;  
Object [] v;  
Object [] [] x;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `x = v;`
- B) `e = (Error []) x;`
- C) `e = v;`
- D) `e = (Error []) v;`
- E) `x = (Object [] []) e;`

### **Versione 24 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] b;  
Exception [] d;  
Boolean [] y;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) d = b;
- B) d = (Exception []) y;
- C) y = (Boolean []) d;
- D) d = (Exception []) b;
- E) d = y;

### **Versione 25 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] [] d;  
Object [] w;  
String [] x;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) x = d;
- B) x = (String []) w;
- C) d = (Object [] []) x;
- D) d = x;
- E) d = w;

### **Versione 26 dell'esercizio 2**

Date le dichiarazioni:

```
class A extends Object {...}  
class B extends A {...}  
class C1 extends A {...}
```

e le dichiarazioni di variabile:

```
C1 b;  
B p;  
A t;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) t = p;
- B) b = t;
- C) p = (B) b;
- D) b = p;
- E) p = t;

### **Versione 27 dell'esercizio 2**

Date le dichiarazioni:

```
Error a;  
Object n;  
Integer z;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `z = (Integer) a;`
- B) `a = z;`
- C) `z = n;`
- D) `a = n;`
- E) `n = a;`

### **Versione 28 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] [] g;  
Error [] v;  
Object [] y;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `v = (Error []) g;`
- B) `v = y;`
- C) `g = v;`
- D) `g = y;`
- E) `y = (Object []) g;`

### **Versione 29 dell'esercizio 2**

Date le dichiarazioni:

```
Integer [] b;  
Object [] d;  
Object [] [] v;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `b = d;`
- B) `d = v;`
- C) `v = b;`
- D) `b = v;`
- E) `b = (Integer []) v;`

### **Versione 1 dell'esercizio 2**

Date le dichiarazioni:

```
class C0 extends Object {...}
class Sub2 extends C0 {...}
class C1 extends C0 {...}
```

e le dichiarazioni di variabile:

```
C0 t;
Sub2 v;
C1 z;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `t = z;`
- B) `z = t;`
- C) `v = (Sub2) z;`
- D) `z = v;`
- E) `z = (C1) v;`

## **Versione 2 dell'esercizio 2**

Date le dichiarazioni:

```
Exception [] b;  
Object [] [] p;  
Object [] w;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `b = p;`
- B) `p = (Object []) [] w;`
- C) `b = (Exception []) p;`
- D) `p = w;`
- E) `p = (Object []) [] b;`

### **Versione 3 dell'esercizio 2**

Date le dichiarazioni:

```
Boolean e;  
Object t;  
Error x;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) e = t;
- B) e = x;
- C) x = (Error) e;
- D) x = e;
- E) t = e;

#### **Versione 4 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] h;  
Object [] [] v;  
Exception [] z;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `h = (Object []) z;`
- B) `z = (Exception []) v;`
- C) `v = (Object [] []) z;`
- D) `z = v;`
- E) `v = z;`

### **Versione 5 dell'esercizio 2**

Date le dichiarazioni:

```
Integer e;  
String v;  
Object y;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `e = v;`
- B) `e = (Integer) y;`
- C) `v = e;`
- D) `e = (Integer) v;`
- E) `v = (String) e;`

### **Versione 6 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] [] c;  
Exception [] p;  
Object [] z;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `c = (Object []) [] p;`
- B) `c = z;`
- C) `p = z;`
- D) `c = p;`
- E) `z = (Object []) c;`

### **Versione 7 dell'esercizio 2**

Date le dichiarazioni:

```
Error e;  
Boolean q;  
Object r;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) r = (Object) e;
- B) e = (Error) q;
- C) q = (Boolean) e;
- D) q = e;
- E) e = r;

### **Versione 8 dell'esercizio 2**

Date le dichiarazioni:

```
class C0 extends Object {...}
class B2 extends C0 {...}
class Sub2 extends C0 {...}
```

e le dichiarazioni di variabile:

```
C0 g;
B2 v;
Sub2 x;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `v = (B2) x;`
- B) `g = (C0) x;`
- C) `x = v;`
- D) `v = x;`
- E) `v = g;`

### **Versione 9 dell'esercizio 2**

Date le dichiarazioni:

```
Integer [] c;  
Object [] d;  
Exception [] p;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `p = d;`
- B) `c = (Integer []) p;`
- C) `p = c;`
- D) `p = (Exception []) d;`
- E) `p = (Exception []) c;`

### **Versione 10 dell'esercizio 2**

Date le dichiarazioni:

```
class A extends Object {...}
class C1 extends A {...}
class Sub1 extends A {...}
```

e le dichiarazioni di variabile:

```
A a;
Sub1 c;
C1 f;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `f = (C1) c;`
- B) `c = f;`
- C) `f = a;`
- D) `c = (Sub1) f;`
- E) `f = (C1) a;`

### **Versione 11 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] [] e;  
Exception [] v;  
Object [] w;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `v = w;`
- B) `v = (Exception []) e;`
- C) `v = e;`
- D) `e = v;`
- E) `w = e;`

### **Versione 12 dell'esercizio 2**

Date le dichiarazioni:

```
Exception [] g;  
Object [] m;  
Error [] p;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `g = (Exception []) p;`
- B) `m = (Object []) p;`
- C) `p = (Error []) g;`
- D) `g = m;`
- E) `p = g;`

### **Versione 13 dell'esercizio 2**

Date le dichiarazioni:

```
class C0 extends Object {...}
class B2 extends C0 {...}
class Sub2 extends C0 {...}
```

e le dichiarazioni di variabile:

```
B2 a;
C0 c;
Sub2 h;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `h = c;`
- B) `h = a;`
- C) `c = h;`
- D) `a = h;`
- E) `h = (Sub2) a;`

### **Versione 14 dell'esercizio 2**

Date le dichiarazioni:

```
Error [] b;  
Object [] e;  
Object [] [] q;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `b = q;`
- B) `e = (Object []) q;`
- C) `b = (Error []) q;`
- D) `b = e;`
- E) `q = b;`

### **Versione 15 dell'esercizio 2**

Date le dichiarazioni:

```
Exception [] d;  
Object [] [] f;  
Object [] q;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `d = q;`
- B) `d = f;`
- C) `f = (Object []) [] d;`
- D) `q = (Object []) f;`
- E) `f = d;`

### **Versione 16 dell'esercizio 2**

Date le dichiarazioni:

```
class A extends Object {...}
class Sub1 extends A {...}
class B2 extends A {...}
```

e le dichiarazioni di variabile:

```
B2 g;
A p;
Sub1 r;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `g = (B2) r;`
- B) `g = r;`
- C) `g = p;`
- D) `r = (Sub1) g;`
- E) `p = r;`

### **Versione 17 dell'esercizio 2**

Date le dichiarazioni:

```
Object b;  
Boolean d;  
Integer y;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `d = y;`
- B) `y = d;`
- C) `d = b;`
- D) `d = (Boolean) y;`
- E) `y = (Integer) b;`

### **Versione 18 dell'esercizio 2**

Date le dichiarazioni:

```
class C0 extends Object {...}
class B extends C0 {...}
class Sub1 extends C0 {...}
```

e le dichiarazioni di variabile:

```
C0 a;
B g;
Sub1 s;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) a = (C0) s;
- B) s = a;
- C) s = g;
- D) s = (Sub1) g;
- E) g = s;

### **Versione 19 dell'esercizio 2**

Date le dichiarazioni:

```
Object [] n;  
Object [] [] t;  
Integer [] v;
```

indicare quale dei seguenti assegnamenti è corretto a tempo di compilazione.

- A) `n = t;`
- B) `v = (Integer []) t;`
- C) `t = v;`
- D) `t = (Object [] []) v;`
- E) `v = n;`

### **Versione 1 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In C le variabili non locali di funzioni e procedure si possono trovare:

- A) sullo stack
- B) nello heap
- C) nella zona statica

ii) Java adotta lo scoping statico?

- A) si
- B) no

### **Versione 2 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Nel primo Fortran esistevano già dei costrutti per allocare memoria a runtime?
  - A) si
  - B) no
  
- ii) Lisp adotta lo scoping statico?
  - A) si
  - B) no

### **Versione 3 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Si può accedere alle variabili non locali di una procedura in tempo costante, indipendentemente da quanti record di attivazione si devono attraversare?
  - A) si
  - B) no
  
- ii) Java ha una implementazione:
  - A) compilata
  - B) interpretata
  - C) mista

### **Versione 4 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Quali forme di polimorfismo supporta C++?
  - A) ad hoc
  - B) per inclusione
  - C) parametrico
  
- ii) Il primo Fortran allocava memoria dinamicamente?
  - A) si
  - B) no

### **Versione 5 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Esistono linguaggi a oggetti senza la gerarchia di tipi?
  - A) si
  - B) no
  
- ii) Nel primo Fortran l'occupazione di memoria di un programma era nota a compile time?
  - A) si
  - B) no

### **Versione 6 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Java adotta lo scoping statico?

- A) si
- B) no

ii) C adotta lo scoping statico?

- A) si
- B) no

### **Versione 7 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) È vero che in C gli int sono lunghi 32 bit?
  - A) si
  - B) no
  - C) dipende
  
- ii) In Java l'ambiente non locale di una classe interna a un metodo si può trovare
  - A) sullo stack
  - B) nello heap
  - C) nella zona statica

### **Versione 8 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) La JVM può caricare classi da host diversi?

- A) si
- B) no

ii) Il C supporta l'equivalenza per nome

- A) sui tipi primitivi
- B) sulle struct

### **Versione 9 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Il C++ ha un garbage collector?
  - A) si
  - B) no
- ii) In un linguaggio dinamicamente tipato una variabile può cambiare il suo tipo durante l'esecuzione del programma?
  - A) si
  - B) no

### **Versione 10 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Nel paradigma logico una chiamata a un predicato può restituire più di un risultato?

- A) si
- B) no

ii) Java è dinamicamente tipato?

- A) si
- B) no

### **Versione 11 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) È vero che nel paradigma funzionale puro non c'è una memoria ma solo un ambiente?

- A) si
- B) no

ii) Un linguaggio senza alcuna forma di cicli (for, while, do-while) può essere computazionalmente completo?

- A) si
- B) no

### **Versione 12 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Nei linguaggi funzionali il valore dei parametri delle funzioni può essere modificato?

- A) si
- B) no

ii) Il C supporta l'equivalenza strutturale

- A) sui tipi primitivi
- B) sulle struct

### **Versione 13 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Quali forme di polimorfismo supporta C++?
  - A) ad hoc
  - B) per inclusione
  - C) parametrico
  
- ii) I parametri IN passati per riferimento
  - A) non possono essere letti prima di essere inizializzati
  - B) non possono essere modificati
  - C) nessuna delle precedenti

### **Versione 14 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Lisp adotta lo scoping statico?

- A) si
- B) no

ii) È vero che nel paradigma funzionale puro non ci sono gli assegnamenti?

- A) si
- B) no

### **Versione 15 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In un linguaggio che supporta solo forme di polimorfismo ad hoc e parametrico e niente record varianti il controllo dei tipi può essere interamente statico?

- A) si
- B) no

ii) La JVM può caricare classi da host diversi?

- A) si
- B) no

### **Versione 16 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Il primo Fortran allocava memoria dinamicamente?
  - A) si
  - B) no
- ii) Quando viene invocata una macro (come le #define in C), viene inserito un record di attivazione nello stack?
  - A) si
  - B) no
  - C) dipende

### **Versione 17 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In una implementazione interpretativa pura l'interprete produce un file con il codice oggetto a partire dal sorgente?

- A) si
- B) no

ii) L'accesso a una variabile non locale nei linguaggi imperativi può essere realizzato in tempo costante, indipendentemente dalla dimensione dello stack?

- A) si
- B) no

### **Versione 18 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Il polimorfismo per inclusione è meno indicato di quello parametrico per la definizione di collezioni di oggetti omogenei?
  - A) si
  - B) no
  
- ii) ML supporta l'equivalenza di tipi
  - A) per nome (name equivalence)
  - B) strutturale (structural equivalence)

### **Versione 19 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In un linguaggio staticamente tipato il controllo dei tipi avviene sempre interamente a tempo di compilazione

- A) si
- B) no

ii) Nelle soluzioni puramente interpretate, l'interprete di un linguaggio L riceve in ingresso sia un programma P scritto in L che gli input di P?

- A) si
- B) no

### **Versione 1 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Nei linguaggi fortemente tipati il type checking avviene tutto a tempo di compilazione?

- A) si
- B) no

ii) Se in Java non si usa il polimorfismo per inclusione, allora tutti i controlli di tipo possono avvenire a tempo di compilazione?

- A) si
- B) no

### **Versione 2 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In un programma che usa record varianti (come le union del C) il controllo dei tipi può essere interamente statico?

- A) si
- B) no

ii) La proprietà caratterizzante del paradigma a oggetti è:

- A) l'encapsulation
- B) la gerarchia di tipi
- C) le interfacce

### **Versione 3 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Il polimorfismo per inclusione è meno indicato di quello parametrico per la definizione di collezioni di oggetti omogenei?

- A) si
- B) no

ii) Un linguaggio fortemente e *\*staticamente\** tipato può avere le union del C (o il costrutto equivalente detto record con varianti di Pascal)?

- A) si
- B) no

### **Versione 4 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Nel primo Fortran esistevano già dei costrutti per allocare memoria a runtime?
  - A) si
  - B) no
  
- ii) Java ha metodi analoghi alla funzione free() di C e C++?
  - A) si
  - B) no

### **Versione 5 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Nei linguaggi fortemente tipati il type checking avviene tutto a tempo di compilazione?

- A) si
- B) no

ii) Il C adotta sempre la structural equivalence tra tipi?

- A) si
- B) no

### **Versione 6 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Il C++ ha un garbage collector?
  - A) si
  - B) no
- ii) In C++ l'ereditarietà multipla può causare un consumo esponenziale di memoria?
  - A) si
  - B) no

### **Versione 7 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Le macro (come ad es. le #define del C) hanno un proprio ambiente locale implementato con un record di attivazione?
  - A) si
  - B) no
  - C) dipende
  
- ii) Un linguaggio fortemente e \*staticamente\* tipato può avere le union del C (o il costrutto equivalente detto record con varianti di Pascal)?
  - A) si
  - B) no

### **Versione 8 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In un linguaggio fortemente tipato il controllo dei tipi avviene sempre interamente a tempo di compilazione

- A) si
- B) no

ii) I parametri IN OUT passati per riferimento

- A) non possono essere letti prima di essere inizializzati
- B) non possono essere modificati
- C) nessuna delle precedenti

### **Versione 9 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Nelle soluzioni puramente interpretate, l'interprete di un linguaggio L riceve in ingresso sia un programma P scritto in L che gli input di P?
  - A) si
  - B) no
  
- ii) Quali forme di polimorfismo supporta Java?
  - A) ad hoc
  - B) per inclusione
  - C) parametrico

### **Versione 10 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) C adotta lo scoping statico?
  - A) si
  - B) no
- ii) In un linguaggio fortemente tipato il controllo dei tipi avviene sempre interamente a tempo di compilazione
  - A) si
  - B) no

### **Versione 11 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) I parametri IN passati per riferimento
  - A) non possono essere letti prima di essere inizializzati
  - B) non possono essere modificati
  - C) nessuna delle precedenti
  
- ii) In un linguaggio fortemente tipato il controllo dei tipi avviene sempre interamente a tempo di compilazione
  - A) si
  - B) no

### **Versione 12 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) In un linguaggio che supporta il polimorfismo per inclusione il controllo dei tipi può essere interamente statico?
  - A) si
  - B) no
  
- ii) ML supporta l'equivalenza di tipi
  - A) per nome (name equivalence)
  - B) strutturale (structural equivalence)

### **Versione 13 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Il primo Fortran allocava memoria dinamicamente?
  - A) si
  - B) no
  
- ii) SQL è un linguaggio di programmazione general purpose?
  - A) si
  - B) no

### **Versione 14 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Le macro (come ad es. le #define del C) hanno un proprio ambiente locale implementato con un record di attivazione?
  - A) si
  - B) no
  - C) dipende
  
- ii) In un linguaggio che supporta solo forme di polimorfismo ad hoc e parametrico e niente record varianti il controllo dei tipi può essere interamente statico?
  - A) si
  - B) no

### **Versione 15 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) In una implementazione compilativa pura il compilatore riceve anche l'input del programma sorgente?
  - A) si
  - B) no
  
- ii) In Java le variabili non locali dei metodi si possono trovare:
  - A) sullo stack
  - B) nello heap
  - C) nella zona statica

### **Versione 16 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) I parametri IN passati per copia
  - A) non possono essere letti prima di essere inizializzati
  - B) non possono essere modificati
  - C) nessuna delle precedenti
  
- ii) Il polimorfismo che permette più controlli a tempo di compilazione è quello
  - A) per inclusione
  - B) parametrico

### **Versione 17 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) I parametri OUT passati per copia
  - A) non possono essere letti prima di essere inizializzati
  - B) non possono essere modificati
  - C) nessuna delle precedenti
  
- ii) In un linguaggio dinamicamente tipato una variabile può cambiare il suo tipo durante l'esecuzione del programma?
  - A) si
  - B) no

### **Versione 18 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) SQL è un linguaggio di programmazione general purpose?
  - A) si
  - B) no
  
- ii) In Java, se l'unico polimorfismo universale usato è quello per inclusione, allora tutti gli eventuali errori di tipo sono segnalati a tempo di compilazione?
  - A) si
  - B) no

### **Versione 19 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In Java le istanze di una classe interna sono sempre associate a una istanza della classe esterna?

- A) si
- B) no

ii) Il Lisp ha un garbage collector?

- A) si
- B) no

### **Versione 1 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) In un linguaggio dinamicamente tipato il controllo dei tipi avviene sempre interamente a tempo di compilazione
  - A) si
  - B) no
  
- ii) Gli attributi statici di una classe Java sono memorizzati nello Heap?
  - A) si
  - B) no

### **Versione 2 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In C++ l'ereditarietà multipla può causare un consumo esponenziale di memoria?

- A) si
- B) no

ii) Nel paradigma funzionale puro si possono usare cicli for e while

- A) si
- B) no

### **Versione 3 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) La differenza tra parametri di ingresso e di uscita svanisce nei linguaggi
  - A) imperativi
  - B) funzionali
  - C) logici
  - D) a oggetti
  
- ii) L'accesso a una variabile non locale nei linguaggi imperativi può essere realizzato in tempo costante, indipendentemente dalla dimensione dello stack?
  - A) si
  - B) no

### **Versione 4 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) HTTP (senza script) è un linguaggio general purpose?
  - A) si
  - B) no
  
- ii) Se in Java non si usa il polimorfismo per inclusione, allora tutti i controlli di tipo possono avvenire a tempo di compilazione?
  - A) si
  - B) no

### **Versione 5 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Esistono linguaggi dove le modalità di passaggio dei parametri vengono determinate solo a run time

- A) si
- B) no

ii) In un linguaggio staticamente tipato il controllo dei tipi avviene sempre interamente a tempo di compilazione

- A) si
- B) no

### **Versione 6 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) I parametri IN passati per copia
  - A) non possono essere letti prima di essere inizializzati
  - B) non possono essere modificati
  - C) nessuna delle precedenti
  
- ii) L'accesso a una variabile non locale nei linguaggi imperativi può essere realizzato in tempo costante, indipendentemente dalla dimensione dello stack?
  - A) si
  - B) no

### **Versione 7 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) La JVM può caricare classi da host diversi?
  - A) si
  - B) no
  
- ii) Il primo Fortran allocava memoria dinamicamente?
  - A) si
  - B) no

### **Versione 8 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Java è dinamicamente tipato?
  - A) si
  - B) no
  
- ii) Il polimorfismo per inclusione è più indicato di quello parametrico per la definizione di collezioni di oggetti omogenei?
  - A) si
  - B) no

### **Versione 9 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Il primo Fortran aveva l'equivalente di una malloc?
  - A) si
  - B) no
  
- ii) La proprietà di invertibilità dei linguaggi logici prende il nome dal fatto che un unico programma può essere usato per calcolare sia una funzione sia la sua inversa?
  - A) si
  - B) no

### **Versione 10 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Java è dinamicamente tipato?
  - A) si
  - B) no
  
- ii) È vero che in C gli int sono lunghi 32 bit?
  - A) si
  - B) no
  - C) dipende

### **Versione 11 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Nei linguaggi con implementazione ibrida compilata/interpretata l'input del programma viene elaborato
  - A) dal compilatore
  - B) dall'interprete
  
- ii) Se A è una istanza di B e B è una istanza di C allora A è una istanza di C?
  - A) si
  - B) no

### **Versione 12 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In un linguaggio che supporta il polimorfismo per inclusione il controllo dei tipi può essere sempre fatto interamente a tempo di compilazione?

- A) si
- B) no

ii) Il primo Fortran aveva un garbage collector?

- A) si
- B) no

### **Versione 13 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Il C ha una implementazione:

- A) compilata
- B) interpretata
- C) mista

ii) Nei linguaggi con implementazione ibrida compilata/interpretata l'input del programma viene elaborato

- A) dal compilatore
- B) dall'interprete

### **Versione 14 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) È vero che in C gli int sono lunghi 32 bit?

- A) si
- B) no
- C) dipende

ii) Il primo Fortran aveva un garbage collector?

- A) si
- B) no

### **Versione 15 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) La JVM può caricare classi da host diversi?

- A) si
- B) no

ii) Il C supporta l'equivalenza per nome

- A) sui tipi primitivi
- B) sulle struct

### **Versione 16 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Java ha una implementazione:

- A) compilata
- B) interpretata
- C) mista

ii) Il polimorfismo per inclusione è più indicato di quello parametrico per la definizione di collezioni di oggetti omogenei?

- A) si
- B) no

### **Versione 17 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Esistono linguaggi fortemente e staticamente tipati dove gli identificatori non vanno necessariamente dichiarati prima dell'uso

- A) si
- B) no

ii) Un linguaggio fortemente e \*staticamente\* tipato può avere le union del C (o il costrutto equivalente detto record con varianti di Pascal)?

- A) si
- B) no

### **Versione 18 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) ML supporta l'equivalenza per nome
  - A) sulle dichiarazioni fatte con "type"
  - B) sulle dichiarazioni fatte con "datatype"
  
- ii) Il C++ ha un garbage collector?
  - A) si
  - B) no

### **Versione 19 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Una lista di oggetti i cui tipi non sono confrontabili tra loro si può definire con il polimorfismo parametrico (templates)?

- A) si
- B) no

ii) Il predicato Prolog member può enumerare i membri di una lista e generare liste?

- A) si
- B) no

### **Versione 20 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Il predicato Prolog member può essere utilizzato sia per verificare se un dato elemento appartiene a una lista sia come “generatore”, cioè per enumerare tutti i membri della lista uno a uno?

- A) si
- B) no

ii) In una implementazione compilativa pura il compilatore riceve anche l'input del programma sorgente?

- A) si
- B) no

### **Versione 21 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In Java le variabili non locali dei metodi si possono trovare:

- A) sullo stack
- B) nello heap
- C) nella zona statica

ii) Il Lisp ha un garbage collector?

- A) si
- B) no

### **Versione 22 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Il C ha una implementazione:
  - A) compilata
  - B) interpretata
  - C) mista
  
- ii) La dimensione degli oggetti nello heap può essere esponenziale nella altezza della gerarchia delle classi (ovvero nel numero di superclassi di una data classe)
  - A) in C
  - B) in C++
  - C) in Java
  - D) mai

### **Versione 23 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In un linguaggio dinamicamente tipato una variabile può cambiare il suo tipo durante l'esecuzione del programma?

- A) si
- B) no

ii) In un linguaggio staticamente tipato il controllo dei tipi avviene sempre interamente a tempo di compilazione

- A) si
- B) no

### **Versione 24 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) È vero che In un linguaggio imperativo, un identificatore x in un assegnamento rappresenta  $\text{env}(x)$ ?
  - A) si
  - B) no
  - C) dipende
  
- ii) Una lista di oggetti i cui tipi non sono confrontabili tra loro si può definire con il polimorfismo parametrico (templates)?
  - A) si
  - B) no

### **Versione 25 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Nei linguaggi fortemente tipati il type checking avviene tutto a tempo di compilazione?

- A) si
- B) no

ii) Java ha metodi analoghi alla funzione free() di C e C++?

- A) si
- B) no

### **Versione 26 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Gli attributi statici di una classe Java sono memorizzati nello Heap?
  - A) si
  - B) no
  
- ii) Se il linguaggio è dinamicamente tipato, allora il tipo di una variabile può cambiare durante l'esecuzione del programma?
  - A) si
  - B) no

### **Versione 27 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Se in Java non si usa il polimorfismo per inclusione, allora tutti i controlli di tipo possono avvenire a tempo di compilazione?
  - A) si
  - B) no
  
- ii) Java ha metodi analoghi alla funzione free() di C e C++?
  - A) si
  - B) no

### **Versione 28 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Gli attributi statici di una classe Java sono memorizzati nello Heap?

- A) si
- B) no

ii) Quali forme di polimorfismo supporta Java?

- A) ad hoc
- B) per inclusione
- C) parametrico

### **Versione 29 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Quali forme di polimorfismo supporta Java?
  - A) ad hoc
  - B) per inclusione
  - C) parametrico
  
- ii) È vero che i linguaggi funzionali non sono computazionalmente completi?
  - A) si
  - B) no

### **Versione 1 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Un tipo di dato astratto è

- A) una classe astratta
- B) una interfaccia
- C) un tipo perfettamente incapsulato

ii) In Java le variabili non locali dei metodi si possono trovare:

- A) sullo stack
- B) nello heap
- C) nella zona statica

### **Versione 2 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) È vero che In un linguaggio funzionale puro, un identificatore x in una espressione rappresenta  $\text{env}(x)$ ?

- A) si
- B) no
- C) dipende

ii) La JVM può caricare classi da host diversi?

- A) si
- B) no

### **Versione 3 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Il polimorfismo per inclusione è meno indicato di quello parametrico per la definizione di collezioni di oggetti omogenei?

- A) si
- B) no

ii) La JVM può caricare classi da host diversi?

- A) si
- B) no

### **Versione 4 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In un linguaggio dinamicamente tipato il controllo dei tipi avviene sempre interamente a tempo di compilazione

- A) si
- B) no

ii) Il C adotta sempre la structural equivalence tra tipi?

- A) si
- B) no

### **Versione 5 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) SQL è un linguaggio di programmazione general purpose?
  - A) si
  - B) no
  
- ii) Il predicato Prolog member può enumerare i membri di una lista e generare liste?
  - A) si
  - B) no

### **Versione 6 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Se A è una istanza di B e B è una istanza di C allora A è una istanza di C?
  - A) si
  - B) no
  
- ii) Esistono linguaggi dove le modalità di passaggio dei parametri non vengono fissate al momento della compilazione
  - A) si
  - B) no

### **Versione 7 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Lisp adotta lo scoping statico?
  - A) si
  - B) no
  
- ii) Gli attributi statici di una classe Java sono memorizzati nello Heap?
  - A) si
  - B) no

### **Versione 8 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Il comando new in Java alloca memoria
  - A) nello stack
  - B) nello heap
  - C) nella zona statica
  
- ii) In una implementazione interpretativa pura l'interprete produce un file con il codice oggetto a partire dal sorgente?
  - A) si
  - B) no

### **Versione 9 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) La proprietà caratterizzante del paradigma a oggetti è:

- A) l'encapsulation
- B) la gerarchia di tipi
- C) le interfacce

ii) L'aliasing consiste nel riferirsi alla stessa locazione con nomi diversi?

- A) si
- B) no

### **Versione 10 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) L'ambiente associa gli identificatori direttamente al loro valore nei linguaggi:
  - A) imperativi
  - B) funzionali
  - C) fortemente tipizzati
  
- ii) È vero che In un linguaggio imperativo, un identificatore x in un assegnamento rappresenta  $\text{env}(x)$ ?
  - A) si
  - B) no
  - C) dipende

### **Versione 11 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Nel paradigma logico una chiamata a un predicato può restituire più di un risultato?

- A) si
- B) no

ii) In un linguaggio che supporta il polimorfismo per inclusione il controllo dei tipi può essere interamente statico?

- A) si
- B) no

### **Versione 12 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Il controllo di correttezza dei downcast richiede controlli a runtime?

- A) si
- B) no
- C) dipende

ii) È vero che nel paradigma funzionale puro non c'è una memoria ma solo un ambiente?

- A) si
- B) no

### **Versione 13 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) I riferimenti che collegano obbligatoriamente ciascuna istanza di una classe interna non statica con una istanza della classe esterna servono per
  - A) accedere all'ambiente non locale dei metodi
  - B) rendere più efficiente il garbage collector
  - C) nessuna delle precedenti
  
- ii) Se il linguaggio è dinamicamente tipato, allora il tipo di una variabile può cambiare durante l'esecuzione del programma?
  - A) si
  - B) no

### **Versione 14 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Un linguaggio puramente funzionale ha i cicli while?
  - A) si
  - B) no
  
- ii) Tra i compiti del supporto a run time c'è la gestione della memoria?
  - A) si
  - B) no

### **Versione 15 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) Una lista di oggetti i cui tipi non sono confrontabili tra loro si può definire con il polimorfismo parametrico (templates)?

- A) si
- B) no

ii) Se il linguaggio è dinamicamente tipato, allora il tipo di una variabile può cambiare durante l'esecuzione del programma?

- A) si
- B) no

### **Versione 16 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) È vero che i linguaggi funzionali non sono computazionalmente completi?
  - A) si
  - B) no
  
- ii) Se implementiamo le liste mediante template, i membri di una lista hanno tutti lo stesso tipo (o nel caso object oriented delle sottoclassi di quel tipo)?
  - A) si
  - B) no

### **Versione 17 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

i) In un programma che usa record varianti (come le union del C) il controllo dei tipi può essere interamente statico?

- A) si
- B) no

ii) Il Lisp supporta la ricorsione?

- A) si
- B) no

### **Versione 18 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) ML supporta l'equivalenza di tipi
  - A) per nome (name equivalence)
  - B) strutturale (structural equivalence)
- ii) Nel paradigma imperativo, l'ambiente non locale di procedure e funzioni si può trovare
  - A) sullo stack
  - B) nello heap
  - C) nella zona statica

### **Versione 19 dell'esercizio 3**

Indicare tutte le risposte vere, separando quelle di domande diverse con un punto e virgola.

- i) Il polimorfismo che permette più controlli a tempo di compilazione è quello
  - A) per inclusione
  - B) parametrico
  
- ii) Nel paradigma funzionale puro si possono usare cicli for e while
  - A) si
  - B) no

### **Versione 1 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[a] = y$ , sapendo che x è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a)) + mem(env(z))$  e  $env(y)$ .

### **Versione 2 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = *y$ , sapendo che  $x$  è un puntatore e che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(env(y) + 4)$ .

**Versione 3 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[3] = \&(y[3])$ , sapendo che x è un vettore e che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(env(y))$ .

#### **Versione 4 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = *y$ , sapendo che  $x$  è un puntatore e che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $env(y) + 2$ .

### **Versione 5 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(a + 2) = y[*p]$ , sapendo che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $env(y)$ .

**Versione 6 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  
 $*(a + z) = \&y.$
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(mem(env(a)))$ .

### **Versione 7 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  
 $*x = *(*y)$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $env(y)$ .

**Versione 8 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[a] = *(*y)$ , sapendo che x è un vettore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(mem(env(p)))$  e  $env(y) + 3$ .

**Versione 9 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast x) = \ast(\ast p + 4)$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $env(y)$ .

#### **Versione 10 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  
 $*x = \&(y[2])$ , sapendo che x è un puntatore e che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + 2$  e  $mem(env(y))$ .

### **Versione 11 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = *y$ , sapendo che  $x$  è un puntatore e che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $env(y) + 5$ .

**Versione 12 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x = *y$ , sapendo che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(mem(env(a)) + mem(env(z)))$ .

**Versione 13 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = *y$ , sapendo che  $x$  è un puntatore e che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(env(y) + 2)$ .

#### **Versione 14 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[2] = \&(y[*p])$ , sapendo che x è un vettore e che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(mem(env(p)))$  e  $env(y)$ .

### **Versione 15 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(*x) = y[*p]$ , sapendo che x è un puntatore e che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + 1$  e  $mem(env(y))$ .

#### **Versione 16 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[a] = \&(y[*p])$ , sapendo che x è un vettore e che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $env(y) + 5$ .

### **Versione 17 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = y$ , sapendo che  $x$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(env(a))$  e  $env(y)$ .

#### **Versione 18 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = *y$ , sapendo che  $x$  è un puntatore e che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a))$  e  $env(y) + mem(env(a))$ .

### **Versione 19 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[3] = \&(y[*p])$ , sapendo che x è un puntatore e che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(env(a))$  e  $env(y) + mem(env(a))$ .

### **Versione 1 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(*x) = y[*p]$ , sapendo che x è un puntatore e che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(mem(env(p)))$  e  $env(y)$ .

**Versione 2 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[a] = *(*p)$ , sapendo che x è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + 1$  e  $env(y)$ .

**Versione 3 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast p + 3) = \ast(\ast p + 3)$ .
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(mem(env(a)) + 1)$ .

#### **Versione 4 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = *(*y)$ , sapendo che  $x$  è un vettore e che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(mem(env(y)))$ .

### **Versione 5 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = y[4]$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(mem(env(y)))$ .

**Versione 6 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x = y[a]$ , sapendo che  $y$  è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + 1$  e  $mem(env(y))$ .

### **Versione 7 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = y[3]$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(mem(env(a)))$ .

**Versione 8 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = *y$ , sapendo che  $x$  è un puntatore e che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(mem(env(y)))$ .

**Versione 9 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[5] = \&(y[a])$ , sapendo che x è un puntatore e che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + 3$  e  $mem(env(y))$ .

#### **Versione 10 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(*x) = y[*p]$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $env(y) + 5$ .

#### **Versione 11 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = y$ , sapendo che x è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(env(y) + mem(env(a)))$ .

### **Versione 12 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(*p) = y[*p]$ , sapendo che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a))$  e  $mem(mem(env(a)))$ .

### **Versione 13 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = *y$ , sapendo che  $x$  è un puntatore e che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(env(a))$  e  $mem(env(y))$ .

#### **Versione 14 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast x) = y[a]$ , sapendo che  $x$  è un puntatore e che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + 4$  e  $env(y) + 4$ .

### **Versione 15 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  
 $*x = *(*y)$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(env(y))$ .

#### **Versione 16 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast x) = \ast(\ast p)$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $env(y)$ .

### **Versione 17 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x = *y$ , sapendo che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(mem(env(p)))$  e  $mem(mem(env(y)))$ .

**Versione 18 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x = \&(y[a])$ , sapendo che  $y$  è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(env(y))$ .

### **Versione 19 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = *(*y)$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(env(y) + mem(mem(env(p))))$ .

### **Versione 1 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = *y$ , sapendo che x è un vettore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(env(y) + mem(mem(env(p))))$ .

### **Versione 2 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = y$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(mem(env(y)))$ .

### **Versione 3 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(*p) = y[*p]$ , sapendo che  $y$  è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a)) + 3$  e  $mem(env(y)) + mem(env(a))$ .

#### **Versione 4 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast x) = \ast(\ast y)$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $env(y) + 4$ .

**Versione 5 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  
 $*(a + 1) = y$ .
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a))$  e  $env(y)$ .

### **Versione 6 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = *y$ , sapendo che  $x$  è un puntatore e che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(mem(env(p)))$  e  $env(y) + mem(mem(env(p)))$ .

### **Versione 7 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[a] = y[*p]$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + 5$  e  $env(y) + mem(mem(env(p)))$ .

**Versione 8 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = *(a + 1)$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a))$  e  $env(y) + 1$ .

**Versione 9 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = y[1]$ , sapendo che  $x$  è un puntatore e che  $y$  è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(mem(env(p)))$  e  $env(y)$ .

**Versione 10 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  
 $*(*p + 3) = y$ .
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(env(y) + mem(mem(env(p))))$ .

### **Versione 11 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast p + 5) = \ast(\ast y)$ , sapendo che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $env(y)$ .

**Versione 12 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  
 $*x = \&y$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $env(y)$ .

### **Versione 13 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast x) = \ast(\ast p + 1)$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $\text{mem}(\text{env}(a)) + 4$  e  $\text{mem}(\text{mem}(\text{env}(y)))$ .

#### **Versione 14 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x = \&(y[*p])$ , sapendo che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a)) + 1$  e  $mem(mem(env(a)) + 1)$ .

**Versione 15 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x = y[a]$ , sapendo che  $y$  è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a)) + 2$  e  $env(y) + 4$ .

#### **Versione 16 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[a] = *y$ , sapendo che  $x$  è un puntatore e che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(env(y))$ .

### **Versione 17 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  
 $*x = y$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a)) + 4$  e  $env(y) + mem(mem(env(p)))$ .

#### **Versione 18 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = *a$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(mem(env(y)))$ .

**Versione 19 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  
 $*(*p + z) = \&y$ .
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(mem(env(y)))$ .

**Versione 20 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[a] = *(*p)$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(mem(env(y)))$ .

**Versione 21 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[2] = \&y$ , sapendo che x è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $env(y) + 2$ .

**Versione 22 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[3] = *(*y)$ , sapendo che x è un vettore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + 5$  e  $mem(mem(env(a)) + 3)$ .

**Versione 23 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[a] = *y$ , sapendo che  $x$  è un puntatore e che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $env(y)$ .

**Versione 24 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = y[1]$ , sapendo che x è un vettore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(env(y))$ .

**Versione 25 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x = y[2]$ , sapendo che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(mem(env(a)) + mem(env(z)))$ .

**Versione 26 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast p + 4) = \ast(\ast y)$ , sapendo che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $env(y)$ .

**Versione 27 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = y[2]$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $env(y) + mem(env(a))$ .

**Versione 28 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = *(*y)$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $env(y) + mem(env(a))$ .

**Versione 29 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = *(*y)$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + 5$  e  $env(y)$ .

### **Versione 1 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x = *y$ , sapendo che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a)) + mem(env(z))$  e  $mem(env(y))$ .

### **Versione 2 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = y[a]$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(env(y))$ .

### **Versione 3 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = y[*p]$ , sapendo che x è un vettore e che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(mem(env(p)))$  e  $mem(env(y) + 3)$ .

#### **Versione 4 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[*p] = y[*p]$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $env(y) + mem(mem(env(p)))$ .

### **Versione 5 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(a + 3) = *(y)$ , sapendo che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a)) + 5$  e  $mem(env(y)) + 2$ .

### **Versione 6 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[4] = *(*p + 1)$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(env(a))$  e  $mem(mem(env(y)))$ .

### **Versione 7 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  
 $*x = y$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(mem(env(a)) + 4)$ .

**Versione 8 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast x) = \ast a$ , sapendo che  $x$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(env(a))$  e  $mem(env(y))$ .

**Versione 9 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[3] = \&y$ , sapendo che  $x$  è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(mem(env(y)))$ .

#### **Versione 10 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(*p) = y[*p]$ , sapendo che y è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $env(y) + mem(env(a))$ .

### **Versione 11 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = *(*p)$ , sapendo che x è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(mem(env(a)) + mem(env(z)))$ .

**Versione 12 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[3] = *y$ , sapendo che  $x$  è un vettore e che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(env(y) + 3)$ .

**Versione 13 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[1] = *y$ , sapendo che  $x$  è un vettore e che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + 1$  e  $mem(mem(env(y)))$ .

#### **Versione 14 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast x) = \ast y$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a))$  e  $env(y) + 5$ .

### **Versione 15 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x = *(*y)$ , sapendo che  $y$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $mem(mem(env(y)))$ .

#### **Versione 16 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $\ast(\ast x) = \&y$ , sapendo che  $x$  è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(a))$  e  $env(y) + mem(mem(env(p)))$ .

**Versione 17 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*(\alpha + z) = *(*p)$ .
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x)$  e  $env(y)$ .

**Versione 18 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $x[a] = y$ , sapendo che x è un vettore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $env(x) + mem(mem(env(p)))$  e  $mem(env(y) + 1)$ .

### **Versione 19 dell'esercizio 4**

- i) Esprimere in termini di mem e env le parti sinistra e destra dell'assegnamento  $*x = y[5]$ , sapendo che x è un puntatore e che y è un puntatore.
- ii) Scrivere un assegnamento le cui parti sinistra e destra corrispondono rispettivamente a  $mem(env(x))$  e  $mem(mem(env(y)))$ .

### Versione 1 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class C1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            m();
        }
        finally {
            System.out.print(2);
        }
    }
    static void m() {
        try {
            System.out.print(3);
            throw( new MyExc1() );
        }
        catch( Error t ) {
            System.out.print(4);
        }
    }
}
```

- A) 132Exception in thread "main" MyExc1
- B) Errore a tempo di compilazione
- C) 132
- D) 1342
- E) Nessuna delle precedenti

## Versione 2 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Error { }
public class A1 {
    public static void main(String [] argv) {
        try {
            q();
            System.out.print(1);
        }
        finally {
            System.out.print(2);
        }
    }
    static void q() {
        try {
            System.out.print(3);
            throw( new MyExc2() );
        }
        catch( Error h ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
            throw( new MyExc1() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 3452Exception in thread "main" MyExc1
- C) 345412
- D) 352
- E) Nessuna delle precedenti

### Versione 3 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class C1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            n();
        }
        catch( MyExc3 c ) {
        }
        catch( Exception v ) {
        }
        finally {
            System.out.print(2);
            throw( new MyExc1() );
        }
    }
    static void n() {
        try {
            System.out.print(3);
            throw( new MyExc3() );
        }
        catch( MyExc3 f ) {
            throw( new MyExc3() );
        }
        catch( MyExc2 y ) {
        }
        finally {
            System.out.print(4);
        }
    }
}
```

- A) 13... (ciclo infinito)
- B) Errore a tempo di compilazione
- C) 1342Exception in thread "main" MyExc1
- D) 13
- E) Nessuna delle precedenti

#### Versione 4 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class A1 {
    public static void main(String [] argv) {
        try {
            p();
        } finally {
            System.out.print(1);
            throw( new MyExc3() );
        }
    }
    static void p() {
        try {
            throw( new MyExc1() );
        }
        catch( Exception y ) {
            System.out.print(2);
        }
        finally {
            throw( new Exception() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 21Exception in thread "main" MyExc3
- C) 221Exception in thread "main" MyExc3
- D) 1Exception in thread "main" MyExc3
- E) Nessuna delle precedenti

### Versione 5 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class A1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            m();
            System.out.print(2);
        }
        catch( MyExc1 x ) {
            throw( new MyExc2() );
        }
        finally {
            throw( new MyExc2() );
        }
    }
    static void m() {
        try {
            throw( new Exception() );
        }
        catch( MyExc1 w ) {
        }
        catch( Exception k ) {
            throw( new MyExc3() );
        }
        finally {
            System.out.print(3);
        }
    }
}
```

- A) 1... (ciclo infinito)
- B) Errore a tempo di compilazione
- C) 13Exception in thread "main" MyExc2
- D) 1Exception in thread "main" MyExc2
- E) Nessuna delle precedenti

### Versione 6 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class A1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            n();
            System.out.print(2);
        }
        catch( MyExc2 k ) {
        }
        catch( MyExc1 g ) {
            System.out.print(3);
        }
        catch( Exception e ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
            throw( new MyExc2() );
        }
    }
    static void n() {
        try {
            System.out.print(6);
            throw( new MyExc3() );
        }
        catch( MyExc2 t ) {
        }
        catch( MyExc3 v ) {
        }
    }
}
```

- A) 162
- B) 1625Exception in thread "main" MyExc2
- C) 1625
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 7 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends MyExc2 { }
public class D1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            m();
            System.out.print(2);
        }
        finally {
            System.out.print(3);
        }
    }
    static void m() {
        try {
            throw( new MyExc1() );
        }
        catch( MyExc3 e ) {
            throw( new MyExc1() );
        }
        catch( Error c ) {
            System.out.print(4);
        }
        finally {
            throw( new MyExc1() );
        }
    }
}
```

- A) 13
- B) Errore a tempo di compilazione
- C) 13Exception in thread "main" MyExc1
- D) 14423
- E) Nessuna delle precedenti

### Versione 8 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class D1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            p();
            System.out.print(2);
        }
        finally {
            System.out.print(3);
            throw( new MyExc2() );
        }
    }
    static void p() {
        try {
            System.out.print(4);
            throw( new MyExc1() );
        }
        catch( MyExc2 u ) {
        }
        catch( Exception k ) {
            System.out.print(5);
        }
        finally {
            System.out.print(6);
        }
    }
}
```

- A) 145623Exception in thread "main" MyExc2
- B) 1463Exception in thread "main" MyExc2
- C) Errore a tempo di compilazione
- D) 1462
- E) Nessuna delle precedenti

### Versione 9 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class D1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            q();
            System.out.print(2);
        }
        catch( MyExc1 v ) {
            throw( new Exception() );
        }
        finally {
            System.out.print(3);
        }
    }
    static void q() {
        try {
            System.out.print(4);
            throw( new MyExc2() );
        }
        catch( MyExc2 y ) {
            throw( new MyExc3() );
        }
        finally {
            throw( new MyExc1() );
        }
    }
}
```

- A) 14... (ciclo infinito)
- B) 143
- C) Errore a tempo di compilazione
- D) 143Exception in thread "main" java.lang.Exception
- E) Nessuna delle precedenti

### Versione 10 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class D1 {
    public static void main(String [] argv) {
        try {
            q();
            System.out.print(1);
        }
        catch( MyExc1 c ) {
        }
        catch( MyExc3 b ) {
        }
        catch( Exception y ) {
            System.out.print(2);
            throw( new MyExc2() );
        }
    }
    static void q() {
        try {
            throw( new MyExc3() );
        }
        catch( MyExc3 f ) {
        }
        finally {
            throw( new MyExc2() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 1
- C) 2Exception in thread "main" MyExc2
- D) 2222222... (ciclo infinito)
- E) Nessuna delle precedenti

### Versione 11 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class A1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            q();
            System.out.print(2);
        }
        catch( MyExc2 z ) {
            System.out.print(3);
            throw( new MyExc2() );
        }
        catch( MyExc1 z ) {
        }
        catch( Exception y ) {
        }
        finally {
            throw( new MyExc1() );
        }
    }
    static void q() {
        try {
        }
        finally {
            System.out.print(4);
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 142Exception in thread "main" MyExc1
- C) 142
- D) 12
- E) Nessuna delle precedenti

### Versione 12 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class C1 {
    public static void main(String [] argv) {
        try {
            m();
            System.out.print(1);
        }
        catch( Exception y ) {
        }
        finally {
            System.out.print(2);
            throw( new MyExc2() );
        }
    }
    static void m() {
        try {
            System.out.print(3);
            throw( new Exception() );
        }
        catch( Exception j ) {
        }
        finally {
            System.out.print(4);
        }
    }
}
```

- A) 3412Exception in thread "main" MyExc2
- B) Errore a tempo di compilazione
- C) 3412
- D) 31
- E) Nessuna delle precedenti

### Versione 13 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class D1 {
    public static void main(String [] argv) {
        try {
            p();
        }
        catch( MyExc3 w ) {
            System.out.print(1);
        }
        finally {
            System.out.print(2);
            throw( new MyExc1() );
        }
    }
    static void p() {
        try {
            throw( new MyExc3() );
        }
        catch( MyExc2 s ) {
        }
        catch( Exception r ) {
            System.out.print(3);
            throw( new MyExc3() );
        }
    }
}
```

- A) 2Exception in thread "main" MyExc1
- B) Errore a tempo di compilazione
- C) 1
- D) 12Exception in thread "main" MyExc1
- E) Nessuna delle precedenti

### Versione 14 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Error { }
public class B1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            q();
            System.out.print(2);
        }
        finally {
            System.out.print(3);
            throw( new MyExc1() );
        }
    }
    static void q() {
        try {
            throw( new Error() );
        }
        catch( Error h ) {
            System.out.print(4);
            throw( new MyExc2() );
        }
        finally {
            System.out.print(5);
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 1453Exception in thread "main" MyExc1
- C) 1444444... (ciclo infinito)
- D) 143Exception in thread "main" MyExc1
- E) Nessuna delle precedenti

### Versione 15 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class C1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            n();
            System.out.print(2);
        }
        catch( Exception x ) {
            System.out.print(3);
        }
        finally {
            throw( new MyExc1() );
        }
    }
    static void n() {
        try {
        }
        catch( MyExc1 j ) {
        }
    }
}
```

- A) 12Exception in thread "main" MyExc1
- B) 123
- C) Errore a tempo di compilazione
- D) 12
- E) Nessuna delle precedenti

### Versione 16 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class D1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            p();
        }
        catch( MyExc1 v ) {
            System.out.print(2);
        }
        finally {
            throw( new MyExc2() );
        }
    }
    static void p() {
        try {
            System.out.print(3);
        }
        catch( MyExc3 w ) {
            System.out.print(4);
        }
        catch( MyExc2 c ) {
            throw( new MyExc2() );
        }
        finally {
            System.out.print(5);
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 13
- C) 1352
- D) 135Exception in thread "main" MyExc2
- E) Nessuna delle precedenti

### Versione 17 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class B1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            m();
        }
        catch( MyExc2 s ) {
            System.out.print(2);
        }
        catch( Error a ) {
            System.out.print(3);
        }
        finally {
            System.out.print(4);
        }
    }
    static void m() {
        try {
            throw( new MyExc1() );
        }
        finally {
            System.out.print(5);
        }
    }
}
```

- A) 15
- B) 154Exception in thread "main" MyExc1
- C) 1534
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 18 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv) {
        try {
            n();
            System.out.print(1);
        }
        catch( Error a ) {
            System.out.print(2);
        }
        finally {
            System.out.print(3);
            throw( new MyExc2() );
        }
    }
    static void n() {
        try {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
        }
    }
}
```

- A) 4513Exception in thread "main" MyExc2
- B) 41
- C) Errore a tempo di compilazione
- D) 45132
- E) Nessuna delle precedenti

### Versione 19 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends Error { }
public class C1 {
    public static void main(String [] argv) {
        try {
            n();
            System.out.print(1);
        }
        finally {
            throw( new MyExc2() );
        }
    }
    static void n() {
        try {
            System.out.print(2);
            throw( new MyExc1() );
        }
        catch( MyExc2 h ) {
            System.out.print(3);
            throw( new MyExc3() );
        }
        catch( Error b ) {
            System.out.print(4);
        }
        finally {
            throw( new MyExc3() );
        }
    }
}
```

- A) 2441Exception in thread "main" MyExc2
- B) Errore a tempo di compilazione
- C) 24Exception in thread "main" MyExc2
- D) 2Exception in thread "main" MyExc2
- E) Nessuna delle precedenti

### Versione 1 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            m();
            System.out.print(1);
        }
        catch( Exception u ) {
            System.out.print(2);
        }
        finally {
            System.out.print(3);
            throw( new MyExc2() );
        }
    }
    static void m()
        throws Exception {
        try {
            System.out.print(4);
        }
        catch( MyExc2 t ) {
            System.out.print(5);
        }
        catch( MyExc1 h ) {
            throw( new MyExc1() );
        }
        catch( Exception b ) {
        }
        finally {
            System.out.print(6);
        }
    }
}
```

- A) 41
- B) 4613Exception in thread "main" MyExc2
- C) 46132
- D) Errore a tempo di compilazione

E) Nessuna delle precedenti

## Versione 2 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
            try {
                n();
            }
            catch( MyExc1 a ) {
            }
            catch( MyExc3 j ) {
                throw( new MyExc2() );
            }
            finally {
                System.out.print(1);
                throw( new Exception() );
            }
        }
    static void n()
        throws Exception {
            try {
                throw( new Exception() );
            }
            catch( MyExc2 t ) {
            }
            catch( MyExc3 h ) {
                System.out.print(2);
            }
            catch( Exception z ) {
                System.out.print(3);
                throw( new MyExc2() );
            }
            finally {
                System.out.print(4);
                throw( new MyExc1() );
            }
        }
}
```

- A) 31Exception in thread "main" java.lang.Exception
- B) Errore a tempo di compilazione

- C) 3431Exception in thread "main" java.lang.Exception
- D) 341Exception in thread "main" java.lang.Exception
- E) Nessuna delle precedenti

### Versione 3 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class B1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            n();
            System.out.print(1);
        }
        catch( MyExc1 z ) {
            System.out.print(2);
        }
        finally {
            System.out.print(3);
            throw( new MyExc3() );
        }
    }
    static void n()
        throws Exception {
        try {
            throw( new MyExc2() );
        }
        catch( MyExc1 c ) {
            throw( new MyExc2() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 213Exception in thread "main" MyExc3
- C) ... (ciclo infinito)
- D) 3Exception in thread "main" MyExc3
- E) Nessuna delle precedenti

#### Versione 4 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class B1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            m();
            System.out.print(2);
        }
        catch( MyExc1 j ) {
            System.out.print(3);
        }
        catch( MyExc3 j ) {
        }
        catch( MyExc2 t ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
        }
    }
    static void m()
        throws Exception {
        try {
            throw( new MyExc2() );
        }
        finally {
            System.out.print(6);
        }
    }
}
```

- A) 16425
- B) 162
- C) Errore a tempo di compilazione
- D) 1645
- E) Nessuna delle precedenti

## Versione 5 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            p();
            System.out.print(1);
        }
        catch( MyExc1 s ) {
            throw( new MyExc2() );
        }
        finally {
            System.out.print(2);
        }
    }
    static void p()
        throws Exception {
        try {
            throw( new MyExc3() );
        }
        catch( MyExc2 g ) {
            System.out.print(3);
        }
        catch( Exception y ) {
            throw( new MyExc2() );
        }
        finally {
            throw( new MyExc3() );
        }
    }
}
```

- A) 2
- B) 3312
- C) 2Exception in thread "main" MyExc3
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 6 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            q();
            System.out.print(1);
        }
        catch( MyExc3 t ) {
            System.out.print(2);
            throw( new MyExc3() );
        }
        finally {
            System.out.print(3);
        }
    }
    static void q()
        throws Exception {
        try {
            throw( new MyExc3() );
        }
        catch( MyExc2 z ) {
        }
        finally {
            System.out.print(4);
        }
    }
}
```

- A) 41
- B) 423Exception in thread "main" MyExc3
- C) Errore a tempo di compilazione
- D) 4222222... (ciclo infinito)
- E) Nessuna delle precedenti

### Versione 7 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            m();
        }
        catch( Exception b ) {
            System.out.print(1);
        }
        finally {
            throw( new MyExc1() );
        }
    }
    static void m()
        throws Exception {
        try {
            System.out.print(2);
            throw( new MyExc2() );
        }
        catch( MyExc2 x ) {
            throw( new Exception() );
        }
    }
}
```

- A) 21Exception in thread "main" MyExc1
- B) Errore a tempo di compilazione
- C) 21
- D) 211
- E) Nessuna delle precedenti

### Versione 8 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            m();
        }
        catch( Exception z ) {
            System.out.print(2);
            throw( new MyExc1() );
        }
        finally {
            System.out.print(3);
        }
    }
    static void m()
        throws Exception {
        try {
            System.out.print(4);
            throw( new Exception() );
        }
        catch( Exception e ) {
            throw( new MyExc2() );
        }
        finally {
            System.out.print(5);
        }
    }
}
```

- A) 143
- B) 14... (ciclo infinito)
- C) Errore a tempo di compilazione
- D) 1453Exception in thread "main" MyExc2
- E) Nessuna delle precedenti

### Versione 9 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class C1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            n();
            System.out.print(2);
        }
        catch( Exception b ) {
            throw( new Exception() );
        }
        finally {
            System.out.print(3);
            throw( new MyExc3() );
        }
    }
    static void n()
        throws Exception {
        try {
            System.out.print(4);
            throw( new MyExc1() );
        }
        catch( Exception j ) {
        }
    }
}
```

- A) 143Exception in thread "main" MyExc3
- B) 1423... (ciclo infinito)
- C) 1423Exception in thread "main" java.lang.Exception
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 10 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            n();
            System.out.print(1);
        }
        catch( MyExc1 s ) {
            System.out.print(2);
            throw( new Exception() );
        }
        catch( Exception g ) {
        }
        finally {
            System.out.print(3);
        }
    }
    static void n()
        throws Exception {
        try {
            throw( new MyExc2() );
        }
        catch( MyExc3 u ) {
            System.out.print(4);
            throw( new MyExc1() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 3
- C) 3Exception in thread "main" MyExc2
- D) 13
- E) Nessuna delle precedenti

### Versione 11 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            m();
        }
        catch( MyExc3 u ) {
            System.out.print(2);
            throw( new Exception() );
        }
        finally {
            System.out.print(3);
            throw( new MyExc3() );
        }
    }
    static void m()
        throws Exception {
        try {
            throw( new MyExc3() );
        }
        catch( Exception b ) {
            System.out.print(4);
            throw( new MyExc2() );
        }
    }
}
```

- A) 123Exception in thread "main" MyExc3
- B) Errore a tempo di compilazione
- C) 1444444... (ciclo infinito)
- D) 12Exception in thread "main" java.lang.Exception
- E) Nessuna delle precedenti

### Versione 12 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            p();
        }
        finally {
            System.out.print(1);
        }
    }
    static void p()
        throws Exception {
        try {
            throw( new MyExc3() );
        }
        catch( MyExc1 e ) {
        }
        catch( MyExc2 x ) {
            System.out.print(2);
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 1
- C) 21
- D) 1Exception in thread "main" MyExc3
- E) Nessuna delle precedenti

### Versione 13 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            q();
        }
        catch( MyExc1 y ) {
            System.out.print(1);
            throw( new Exception() );
        }
        catch( MyExc3 e ) {
        }
        catch( MyExc2 s ) {
            System.out.print(2);
        }
    }
    static void q()
        throws Exception {
        try {
            System.out.print(3);
            throw( new MyExc2() );
        }
        catch( MyExc1 t ) {
        }
        catch( MyExc2 h ) {
            throw( new MyExc2() );
        }
        finally {
            throw( new Exception() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 3Exception in thread "main" java.lang.Exception
- C) 3... (ciclo infinito)
- D) 32
- E) Nessuna delle precedenti

### Versione 14 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class B1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            p();
            System.out.print(2);
        }
        catch( Exception j ) {
            throw( new MyExc1() );
        }
        finally {
            System.out.print(3);
        }
    }
    static void p()
        throws Exception {
        try {
            throw( new Exception() );
        }
        catch( MyExc2 d ) {
        }
        catch( MyExc3 b ) {
            throw( new MyExc2() );
        }
        catch( MyExc1 v ) {
            System.out.print(4);
        }
    }
}
```

- A) 1Exception in thread "main" MyExc1
- B) Errore a tempo di compilazione
- C) 1... (ciclo infinito)
- D) 13Exception in thread "main" MyExc1
- E) Nessuna delle precedenti

### Versione 15 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            n();
        }
        catch( MyExc3 a ) {
        }
        finally {
            System.out.print(1);
        }
    }
    static void n()
        throws Exception {
        try {
            System.out.print(2);
            throw( new MyExc2() );
        }
        catch( MyExc1 f ) {
            System.out.print(3);
        }
        finally {
            throw( new Exception() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 231Exception in thread "main" java.lang.Exception
- C) 21Exception in thread "main" java.lang.Exception
- D) 21
- E) Nessuna delle precedenti

### Versione 16 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            m();
            System.out.print(1);
        }
        catch( MyExc3 y ) {
            System.out.print(2);
        }
        catch( MyExc1 f ) {
        }
        finally {
            System.out.print(3);
        }
    }
    static void m()
        throws Exception {
        try {
            throw( new MyExc1() );
        }
        catch( MyExc3 g ) {
            throw( new Exception() );
        }
        finally {
            System.out.print(4);
        }
    }
}
```

- A) 41
- B) Errore a tempo di compilazione
- C) 43
- D) 413
- E) Nessuna delle precedenti

### Versione 17 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            p();
            System.out.print(2);
        }
        catch( Exception g ) {
        }
        finally {
            System.out.print(3);
        }
    }
    static void p()
        throws Exception {
        try {
            throw( new MyExc1() );
        }
        catch( MyExc3 v ) {
        }
        finally {
            System.out.print(4);
        }
    }
}
```

- A) 142
- B) Errore a tempo di compilazione
- C) 143
- D) 1423
- E) Nessuna delle precedenti

### Versione 18 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            p();
        }
        finally {
            System.out.print(1);
            throw( new MyExc3() );
        }
    }
    static void p()
        throws Exception {
        try {
        }
        catch( MyExc2 b ) {
            throw( new MyExc2() );
        }
        finally {
            throw( new MyExc3() );
        }
    }
}
```

- A) 1Exception in thread "main" MyExc3
- B) ... (ciclo infinito)
- C) Errore a tempo di compilazione
- D)
- E) Nessuna delle precedenti

### Versione 19 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            n();
        }
        catch( MyExc2 e ) {
            throw( new Exception() );
        }
        finally {
            System.out.print(2);
        }
    }
    static void n()
        throws Exception {
        try {
            throw( new MyExc2() );
        }
        catch( MyExc1 g ) {
            System.out.print(3);
        }
        catch( Exception t ) {
        }
        finally {
            System.out.print(4);
        }
    }
}
```

- A) 1342
- B) Errore a tempo di compilazione
- C) 14
- D) 142Exception in thread "main" java.lang.Exception
- E) Nessuna delle precedenti

### Versione 1 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class C1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            m();
        }
        catch( Exception k ) {
            System.out.print(2);
        }
        catch( MyExc2 t ) {
        }
        finally {
            throw( new MyExc1() );
        }
    }
    static void m()
        throws Exception {
        try {
            throw( new Exception() );
        }
        catch( MyExc2 i ) {
            System.out.print(3);
            throw( new MyExc3() );
        }
        catch( MyExc1 y ) {
            throw( new MyExc2() );
        }
        catch( Exception e ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
            throw( new MyExc2() );
        }
    }
}
```

- A) 1452Exception in thread "main" MyExc1

- B) Errore a tempo di compilazione
- C) 14
- D) 145342
- E) Nessuna delle precedenti

## Versione 2 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class C1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            n();
        }
    }
    static void n()
        throws Exception {
        try {
            System.out.print(2);
        }
        catch( MyExc1 r ) {
        }
        catch( MyExc3 e ) {
            System.out.print(3);
        }
        finally {
            System.out.print(4);
            throw( new MyExc3() );
        }
    }
}
```

- A) 124Exception in thread "main" MyExc3
- B) 12
- C) 1243
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 3 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            p();
            System.out.print(2);
        }
        catch( MyExc3 v ) {
            throw( new MyExc3() );
        }
        catch( MyExc2 u ) {
            throw( new MyExc1() );
        }
        catch( MyExc1 y ) {
        }
        finally {
            throw( new MyExc2() );
        }
    }
    static void p()
        throws Exception {
        try {
            throw( new MyExc3() );
        }
        catch( MyExc1 v ) {
            System.out.print(3);
            throw( new MyExc1() );
        }
        catch( MyExc2 i ) {
        }
        catch( MyExc3 r ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
            throw( new MyExc1() );
        }
    }
}
```

}

- A) 142
- B) Errore a tempo di compilazione
- C) 145Exception in thread "main" MyExc2
- D) 15Exception in thread "main" MyExc2
- E) Nessuna delle precedenti

#### Versione 4 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            n();
            System.out.print(2);
        }
        catch( MyExc2 b ) {
            System.out.print(3);
        }
        catch( Exception r ) {
            System.out.print(4);
        }
        catch( MyExc1 s ) {
            System.out.print(5);
        }
    }
    static void n()
        throws Exception {
        try {
            throw( new MyExc1() );
        }
        catch( MyExc1 b ) {
            System.out.print(6);
            throw( new MyExc1() );
        }
        finally {
            System.out.print(7);
            throw( new MyExc1() );
        }
    }
}
```

- A) 1666666... (ciclo infinito)
- B) Errore a tempo di compilazione
- C) 1674
- D) 165

E) Nessuna delle precedenti

### Versione 5 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            q();
            System.out.print(1);
        }
        catch( MyExc2 s ) {
            System.out.print(2);
        }
    }
    static void q()
        throws Exception {
        try {
            System.out.print(3);
            throw( new Exception() );
        }
        catch( Exception c ) {
            System.out.print(4);
        }
        catch( MyExc3 c ) {
            System.out.print(5);
            throw( new MyExc3() );
        }
        finally {
            throw( new MyExc2() );
        }
    }
}
```

- A) 3421
- B) 342
- C) Errore a tempo di compilazione
- D) 341
- E) Nessuna delle precedenti

### Versione 6 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            p();
            System.out.print(2);
        }
        catch( MyExc3 f ) {
            System.out.print(3);
        }
        catch( MyExc2 y ) {
            System.out.print(4);
        }
        catch( Exception r ) {
        }
        finally {
            System.out.print(5);
            throw( new MyExc1() );
        }
    }
    static void p()
        throws Exception {
        try {
        }
        catch( Exception i ) {
        }
        catch( MyExc2 t ) {
            System.out.print(6);
        }
        finally {
            throw( new MyExc1() );
        }
    }
}
```

- A) 12
- B) 15Exception in thread "main" MyExc1
- C) 125

D) Errore a tempo di compilazione

E) Nessuna delle precedenti

### Versione 7 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class B1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            m();
            System.out.print(1);
        }
        finally {
            System.out.print(2);
            throw( new MyExc1() );
        }
    }
    static void m()
        throws Exception {
        try {
            System.out.print(3);
            throw( new MyExc1() );
        }
        catch( Exception z ) {
            System.out.print(4);
        }
        catch( MyExc1 b ) {
            System.out.print(5);
        }
    }
}
```

- A) 3412Exception in thread "main" MyExc1
- B) 3512Exception in thread "main" MyExc1
- C) 351
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 8 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            m();
        }
    }
    static void m()
        throws Exception {
        try {
            System.out.print(2);
        }
        catch( MyExc2 d ) {
            System.out.print(3);
        }
        catch( Exception h ) {
            System.out.print(4);
        }
        finally {
            throw( new Exception() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 12
- C) 124
- D) 12Exception in thread "main" java.lang.Exception
- E) Nessuna delle precedenti

### Versione 9 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            m();
            System.out.print(1);
        }
        finally {
            System.out.print(2);
        }
    }
    static void m()
        throws Exception {
        try {
            throw( new Exception() );
        }
        catch( Exception a ) {
            System.out.print(3);
        }
        catch( MyExc3 c ) {
        }
        catch( MyExc2 j ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
            throw( new MyExc1() );
        }
    }
}
```

- A) 352Exception in thread "main" MyExc1
- B) 31
- C) Errore a tempo di compilazione
- D) 35312
- E) Nessuna delle precedenti

### Versione 10 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            p();
            System.out.print(2);
        }
        catch( MyExc1 x ) {
            System.out.print(3);
        }
        catch( MyExc3 w ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
        }
    }
    static void p()
        throws Exception {
        try {
            System.out.print(6);
            throw( new MyExc3() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 164
- C) 1645
- D) 16425
- E) Nessuna delle precedenti

### Versione 11 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            m();
        }
    }
    static void m()
        throws Exception {
        try {
            System.out.print(2);
            throw( new MyExc2() );
        }
        catch( MyExc2 z ) {
            System.out.print(3);
        }
        catch( MyExc1 s ) {
            System.out.print(4);
        }
        catch( Exception d ) {
            System.out.print(5);
        }
        finally {
            System.out.print(6);
            throw( new MyExc3() );
        }
    }
}
```

- A) 123
- B) 12365
- C) 1236Exception in thread "main" MyExc3
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 12 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            p();
        }
        catch( Exception f ) {
            System.out.print(1);
        }
        finally {
            System.out.print(2);
        }
    }
    static void p()
        throws Exception {
        try {
            throw( new MyExc2() );
        }
        catch( Exception y ) {
            System.out.print(3);
        }
        catch( MyExc1 y ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
            throw( new MyExc1() );
        }
    }
}
```

- A) 3532
- B) 3512
- C) 52
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 13 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            n();
        }
        finally {
            System.out.print(1);
        }
    }
    static void n()
        throws Exception {
        try {
            System.out.print(2);
            throw( new MyExc2() );
        }
        catch( Exception w ) {
            System.out.print(3);
        }
        catch( MyExc1 g ) {
        }
        finally {
            System.out.print(4);
        }
    }
}
```

- A) 24
- B) Errore a tempo di compilazione
- C) 241Exception in thread "main" MyExc2
- D) 2341
- E) Nessuna delle precedenti

### Versione 14 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class B1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            m();
        }
        catch( Exception d ) {
            System.out.print(2);
        }
        catch( MyExc3 j ) {
            System.out.print(3);
        }
        catch( MyExc1 h ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
            throw( new MyExc3() );
        }
    }
    static void m()
        throws Exception {
        try {
            System.out.print(6);
            throw( new MyExc3() );
        }
        catch( MyExc2 s ) {
            System.out.print(7);
            throw( new Exception() );
        }
        catch( MyExc1 i ) {
            System.out.print(8);
        }
        finally {
            throw( new MyExc1() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 164
- C) 1625Exception in thread "main" MyExc3
- D) 16252
- E) Nessuna delle precedenti

### Versione 15 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class C1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            p();
            System.out.print(2);
        }
        catch( MyExc1 r ) {
        }
        catch( Exception u ) {
            System.out.print(3);
        }
        finally {
            throw( new MyExc3() );
        }
    }
    static void p()
        throws Exception {
        try {
            System.out.print(4);
            throw( new MyExc1() );
        }
        catch( Exception f ) {
            System.out.print(5);
        }
        catch( MyExc1 z ) {
        }
        catch( MyExc3 t ) {
            System.out.print(6);
        }
    }
}
```

- A) 1452Exception in thread "main" MyExc3
- B) 142Exception in thread "main" MyExc3
- C) 142
- D) Errore a tempo di compilazione

E) Nessuna delle precedenti

### Versione 16 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            p();
            System.out.print(2);
        }
        catch( MyExc1 s ) {
        }
        catch( Exception w ) {
            System.out.print(3);
            throw( new Exception() );
        }
    }
    static void p()
        throws Exception {
        try {
            System.out.print(4);
            throw( new MyExc1() );
        }
        catch( MyExc3 v ) {
            System.out.print(5);
        }
        catch( Exception g ) {
            throw( new Exception() );
        }
        catch( MyExc1 v ) {
            System.out.print(6);
        }
        finally {
            System.out.print(7);
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 14672
- C) 1473Exception in thread "main" java.lang.Exception

- D) 1462
- E) Nessuna delle precedenti

### Versione 17 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            m();
            System.out.print(2);
        }
        catch( MyExc3 y ) {
            System.out.print(3);
        }
        catch( MyExc2 x ) {
        }
        finally {
            System.out.print(4);
        }
    }
    static void m()
        throws Exception {
        try {
            System.out.print(5);
            throw( new Exception() );
        }
        catch( Exception c ) {
            System.out.print(6);
        }
        catch( MyExc1 v ) {
            System.out.print(7);
        }
        catch( MyExc3 z ) {
        }
        finally {
            System.out.print(8);
            throw( new MyExc2() );
        }
    }
}
```

- A) Errore a tempo di compilazione

- B) 15684
- C) 1562
- D) 156824
- E) Nessuna delle precedenti

### Versione 18 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            n();
            System.out.print(2);
        }
        catch( MyExc1 t ) {
            System.out.print(3);
        }
        catch( MyExc2 w ) {
        }
        catch( Exception r ) {
        }
        finally {
            System.out.print(4);
        }
    }
    static void n()
        throws Exception {
        try {
            System.out.print(5);
            throw( new MyExc1() );
        }
        catch( MyExc1 e ) {
            throw( new MyExc3() );
        }
    }
}
```

- A) 15324
- B) Errore a tempo di compilazione
- C) 154
- D) 1534
- E) Nessuna delle precedenti

### Versione 19 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class B1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            n();
            System.out.print(1);
        }
        catch( MyExc2 u ) {
            throw( new MyExc3() );
        }
        finally {
            System.out.print(2);
        }
    }
    static void n()
        throws Exception {
        try {
            throw( new MyExc1() );
        }
        catch( Exception w ) {
            System.out.print(3);
        }
        catch( MyExc3 t ) {
        }
        catch( MyExc1 e ) {
            throw( new Exception() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 312
- C) 2
- D) 2Exception in thread "main" java.lang.Exception
- E) Nessuna delle precedenti

### Versione 20 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            n();
        }
        catch( Exception d ) {
        }
        finally {
            System.out.print(2);
            throw( new MyExc1() );
        }
    }
    static void n()
        throws Exception {
        try {
            System.out.print(3);
            throw( new Exception() );
        }
        catch( MyExc3 r ) {
            System.out.print(4);
        }
        catch( Exception v ) {
            System.out.print(5);
        }
        catch( MyExc1 b ) {
            System.out.print(6);
        }
        finally {
            System.out.print(7);
        }
    }
}
```

- A) 13572
- B) 13572Exception in thread "main" MyExc1
- C) Errore a tempo di compilazione

D) 135

E) Nessuna delle precedenti

### Versione 21 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            n();
        }
        finally {
            System.out.print(2);
        }
    }
    static void n()
        throws Exception {
        try {
            System.out.print(3);
            throw( new MyExc2() );
        }
        catch( Exception k ) {
            System.out.print(4);
            throw( new MyExc1() );
        }
        catch( MyExc2 t ) {
            System.out.print(5);
        }
        catch( MyExc1 i ) {
            throw( new Exception() );
        }
        finally {
            throw( new MyExc3() );
        }
    }
}
```

- A) 135
- B) 13444444... (ciclo infinito)
- C) Errore a tempo di compilazione
- D) 1342Exception in thread "main" MyExc3
- E) Nessuna delle precedenti

## Versione 22 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class C1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            p();
            System.out.print(2);
        }
        catch( MyExc3 w ) {
            System.out.print(3);
        }
        finally {
            System.out.print(4);
        }
    }
    static void p()
        throws Exception {
        try {
            System.out.print(5);
            throw( new MyExc3() );
        }
        catch( Exception z ) {
            throw( new MyExc2() );
        }
        catch( MyExc2 g ) {
            throw( new Exception() );
        }
        catch( MyExc1 c ) {
            System.out.print(6);
            throw( new Exception() );
        }
    }
}
```

- A) 153
- B) 154Exception in thread "main" MyExc2
- C) 1534
- D) Errore a tempo di compilazione

E) Nessuna delle precedenti

### Versione 23 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class C1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            m();
        }
        catch( Exception w ) {
            System.out.print(1);
        }
        catch( MyExc1 h ) {
        }
        catch( MyExc3 a ) {
        }
        finally {
            throw( new MyExc2() );
        }
    }
    static void m()
        throws Exception {
        try {
        }
        catch( MyExc2 r ) {
            System.out.print(2);
        }
        catch( MyExc1 h ) {
            System.out.print(3);
        }
        catch( Exception i ) {
            throw( new Exception() );
        }
    }
}
```

- A) Exception in thread "main" MyExc2
- B)
- C) 1
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 24 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            p();
        }
        finally {
            System.out.print(2);
        }
    }
    static void p()
        throws Exception {
        try {
            System.out.print(3);
            throw( new MyExc1() );
        }
        catch( Exception e ) {
        }
        catch( MyExc3 v ) {
        }
        catch( MyExc1 f ) {
            System.out.print(4);
        }
    }
}
```

- A) 134
- B) 1342
- C) 132
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 25 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class C1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            p();
        }
        catch( MyExc1 f ) {
            System.out.print(2);
        }
        catch( Exception y ) {
        }
        catch( MyExc3 g ) {
        }
        finally {
            throw( new MyExc3() );
        }
    }
    static void p()
        throws Exception {
        try {
            System.out.print(3);
            throw( new Exception() );
        }
        catch( Exception u ) {
        }
        catch( MyExc3 d ) {
            System.out.print(4);
        }
        catch( MyExc1 x ) {
        }
        finally {
            System.out.print(5);
            throw( new Exception() );
        }
    }
}
```

- A) Errore a tempo di compilazione

- B) 1352
- C) 13
- D) 135Exception in thread "main" MyExc3
- E) Nessuna delle precedenti

### Versione 26 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class D1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            m();
            System.out.print(2);
        }
        catch( Exception y ) {
            System.out.print(3);
        }
        catch( MyExc2 c ) {
            System.out.print(4);
        }
    }
    static void m()
        throws Exception {
        try {
            throw( new Exception() );
        }
        finally {
            System.out.print(5);
            throw( new MyExc2() );
        }
    }
}
```

- A) 153
- B) 153Exception in thread "main" java.lang.Exception
- C) Errore a tempo di compilazione
- D) 154
- E) Nessuna delle precedenti

### Versione 27 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class C1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            p();
            System.out.print(2);
        }
        catch( Exception d ) {
            System.out.print(3);
        }
        catch( MyExc2 c ) {
            System.out.print(4);
        }
        catch( MyExc1 u ) {
        }
        finally {
            System.out.print(5);
            throw( new MyExc2() );
        }
    }
    static void p()
        throws Exception {
        try {
            throw( new Exception() );
        }
        catch( MyExc2 z ) {
        }
        catch( MyExc1 v ) {
        }
        finally {
            System.out.print(6);
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 1635Exception in thread "main" MyExc2
- C) 162

D) 16325Exception in thread "main" MyExc2

E) Nessuna delle precedenti

### Versione 28 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            q();
        }
    }
    static void q()
        throws Exception {
        try {
            throw( new Exception() );
        }
        catch( Exception h ) {
            System.out.print(2);
            throw( new MyExc2() );
        }
        finally {
            System.out.print(3);
        }
    }
}
```

- A) 1222222... (ciclo infinito)
- B) 123Exception in thread "main" MyExc2
- C) Errore a tempo di compilazione
- D) 12Exception in thread "main" MyExc2
- E) Nessuna delle precedenti

### Versione 29 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv)
        throws Exception {
        try {
            System.out.print(1);
            p();
        }
        catch( MyExc1 y ) {
        }
        catch( MyExc2 k ) {
        }
        catch( MyExc3 g ) {
            System.out.print(2);
        }
        finally {
            System.out.print(3);
        }
    }
    static void p()
        throws Exception {
        try {
            System.out.print(4);
            throw( new Exception() );
        }
        catch( MyExc3 h ) {
            System.out.print(5);
            throw( new Exception() );
        }
        catch( Exception t ) {
            System.out.print(6);
            throw( new MyExc1() );
        }
        catch( MyExc1 a ) {
            System.out.print(7);
            throw( new Exception() );
        }
        finally {
            System.out.print(8);
        }
    }
}
```

}  
}

- A) 14683
- B) 146
- C) Errore a tempo di compilazione
- D) 14666666... (ciclo infinito)
- E) Nessuna delle precedenti

### Versione 1 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv) {
        try {
            m();
        }
        catch( MyExc2 r ) {
        }
        catch( MyExc1 z ) {
            System.out.print(1);
        }
        catch( Error k ) {
            System.out.print(2);
        }
    }
    static void m() {
        try {
            throw( new MyExc1() );
        }
        catch( MyExc3 c ) {
            System.out.print(3);
        }
        catch( MyExc2 a ) {
            System.out.print(4);
            throw( new MyExc2() );
        }
        catch( Error i ) {
            throw( new Error() );
        }
        finally {
            throw( new MyExc1() );
        }
    }
}
```

- A) 1Exception in thread "main" Error
- B) 1
- C) Errore a tempo di compilazione
- D) ... (ciclo infinito)

E) Nessuna delle precedenti

## Versione 2 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class D1 {
    public static void main(String [] argv) {
        try {
            q();
        }
        catch( MyExc3 v ) {
            throw( new MyExc1() );
        }
        catch( Exception j ) {
            throw( new Exception() );
        }
    }
    static void q() {
        try {
            throw( new Exception() );
        }
        catch( MyExc2 g ) {
            System.out.print(1);
        }
        finally {
            System.out.print(2);
        }
    }
}
```

- A) 2Exception in thread "main" java.lang.Exception
- B) 2
- C) 2... (ciclo infinito)
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 3 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv) {
        try {
            q();
        }
        catch( MyExc2 i ) {
        }
    }
    static void q() {
        try {
            System.out.print(1);
        }
        catch( MyExc2 h ) {
            System.out.print(2);
        }
        catch( Exception d ) {
        }
        finally {
            System.out.print(3);
            throw( new Exception() );
        }
    }
}
```

- A) 1
- B) 13Exception in thread "main" java.lang.Exception
- C) Errore a tempo di compilazione
- D) 13
- E) Nessuna delle precedenti

#### Versione 4 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class D1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            q();
        }
        catch( MyExc2 v ) {
            System.out.print(2);
        }
        catch( MyExc1 s ) {
            throw( new Exception() );
        }
        catch( Exception z ) {
            System.out.print(3);
            throw( new MyExc2() );
        }
    }
    static void q() {
        try {
            System.out.print(4);
            throw( new Exception() );
        }
        finally {
            System.out.print(5);
        }
    }
}
```

- A) 145
- B) 1453Exception in thread "main" MyExc2
- C) Errore a tempo di compilazione
- D) 14532
- E) Nessuna delle precedenti

### Versione 5 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends Error { }
public class A1 {
    public static void main(String [] argv) {
        try {
            p();
            System.out.print(1);
        }
        catch( MyExc1 s ) {
            System.out.print(2);
        }
        catch( MyExc3 b ) {
        }
        catch( Error g ) {
            System.out.print(3);
        }
        finally {
            System.out.print(4);
        }
    }
    static void p() {
        try {
            throw( new MyExc2() );
        }
        finally {
            System.out.print(5);
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 5314
- C) 54Exception in thread "main" MyExc2
- D) 51
- E) Nessuna delle precedenti

### Versione 6 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class A1 {
    public static void main(String [] argv) {
        try {
            q();
            System.out.print(1);
        }
        catch( MyExc3 v ) {
        }
        catch( MyExc1 r ) {
            System.out.print(2);
        }
        finally {
            System.out.print(3);
            throw( new MyExc3() );
        }
    }
    static void q() {
        try {
            System.out.print(4);
            throw( new MyExc2() );
        }
        catch( MyExc2 h ) {
            System.out.print(5);
        }
        catch( MyExc1 z ) {
            throw( new MyExc3() );
        }
        catch( Exception t ) {
        }
    }
}
```

- A) 4513Exception in thread "main" MyExc3
- B) 4513
- C) Errore a tempo di compilazione
- D) 451
- E) Nessuna delle precedenti

### Versione 7 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Exception { }
public class C1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            m();
        }
        catch( MyExc2 y ) {
        }
        catch( MyExc1 s ) {
            throw( new MyExc1() );
        }
        finally {
            throw( new MyExc1() );
        }
    }
    static void m() {
        try {
            System.out.print(2);
        }
        catch( MyExc1 h ) {
            System.out.print(3);
            throw( new Exception() );
        }
        catch( MyExc3 i ) {
            throw( new Exception() );
        }
        catch( Exception c ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
            throw( new MyExc1() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 12534... (ciclo infinito)
- C) 125Exception in thread "main" MyExc1

D) 12

E) Nessuna delle precedenti

### Versione 8 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            n();
        }
        catch( MyExc1 f ) {
        }
        finally {
            throw( new MyExc1() );
        }
    }
    static void n() {
        try {
        }
        finally {
            System.out.print(2);
        }
    }
}
```

- A) 1
- B) Errore a tempo di compilazione
- C) 12Exception in thread "main" MyExc1
- D) 12
- E) Nessuna delle precedenti

### Versione 9 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends MyExc2 { }
public class C1 {
    public static void main(String [] argv) {
        try {
            m();
            System.out.print(1);
        }
        catch( Error v ) {
            System.out.print(2);
        }
        finally {
            throw( new MyExc1() );
        }
    }
    static void m() {
        try {
            System.out.print(3);
        }
        catch( Error f ) {
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 31
- C) 31Exception in thread "main" MyExc1
- D) 312
- E) Nessuna delle precedenti

### Versione 10 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends Error { }
public class B1 {
    public static void main(String [] argv) {
        try {
            n();
            System.out.print(1);
        }
        finally {
            System.out.print(2);
        }
    }
    static void n() {
        try {
            throw( new Error() );
        }
        catch( MyExc1 u ) {
            System.out.print(3);
        }
        catch( Error e ) {
            throw( new MyExc2() );
        }
    }
}
```

- A) 2
- B) 2Exception in thread "main" MyExc2
- C) ... (ciclo infinito)
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 11 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            m();
        }
        finally {
            System.out.print(2);
        }
    }
    static void m() {
        try {
            throw( new MyExc1() );
        }
        catch( MyExc1 h ) {
            System.out.print(3);
        }
        finally {
            throw( new MyExc3() );
        }
    }
}
```

- A) Errore a tempo di compilazione
- B) 1332
- C) 13
- D) 132Exception in thread "main" MyExc3
- E) Nessuna delle precedenti

### Versione 12 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends Error { }
public class D1 {
    public static void main(String [] argv) {
        try {
            p();
            System.out.print(1);
        }
        catch( MyExc1 h ) {
            System.out.print(2);
        }
        catch( MyExc3 x ) {
            System.out.print(3);
            throw( new MyExc1() );
        }
        finally {
            throw( new Error() );
        }
    }
    static void p() {
        try {
            throw( new MyExc3() );
        }
        finally {
            System.out.print(4);
        }
    }
}
```

- A) 43Exception in thread "main" Error
- B) Errore a tempo di compilazione
- C) 41
- D) 432Exception in thread "main" Error
- E) Nessuna delle precedenti

### Versione 13 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends Error { }
class MyExc3 extends MyExc2 { }
public class C1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            p();
        }
        catch( Error h ) {
            System.out.print(2);
        }
        finally {
            throw( new Error() );
        }
    }
    static void p() {
        try {
            System.out.print(3);
            throw( new MyExc2() );
        }
        catch( MyExc2 s ) {
        }
        catch( Error z ) {
            System.out.print(4);
            throw( new MyExc3() );
        }
        finally {
            throw( new MyExc3() );
        }
    }
}
```

- A) 132
- B) 13
- C) 13Exception in thread "main" Error
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 14 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends Exception { }
public class C1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            m();
            System.out.print(2);
        }
        catch( MyExc2 x ) {
            throw( new MyExc2() );
        }
        catch( Exception a ) {
        }
        finally {
            System.out.print(3);
        }
    }
    static void m() {
        try {
            System.out.print(4);
            throw( new MyExc2() );
        }
        finally {
            System.out.print(5);
        }
    }
}
```

- A) 1453Exception in thread "main" MyExc2
- B) Errore a tempo di compilazione
- C) 1452
- D) 145... (ciclo infinito)
- E) Nessuna delle precedenti

### Versione 15 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends Exception { }
class MyExc3 extends MyExc2 { }
public class B1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            q();
            System.out.print(2);
        }
        catch( MyExc1 u ) {
        }
        catch( MyExc2 y ) {
            System.out.print(3);
        }
        catch( Exception f ) {
        }
        finally {
            System.out.print(4);
            throw( new MyExc1() );
        }
    }
    static void q() {
        try {
            System.out.print(5);
            throw( new Exception() );
        }
        catch( MyExc1 i ) {
        }
        catch( MyExc3 r ) {
            System.out.print(6);
            throw( new MyExc1() );
        }
        catch( Exception d ) {
            System.out.print(7);
        }
        finally {
            System.out.print(8);
        }
    }
}
```

A) 157824

- B) 1572
- C) Errore a tempo di compilazione
- D) 157824Exception in thread "main" MyExc1
- E) Nessuna delle precedenti

### Versione 16 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class B1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            n();
            System.out.print(2);
        }
        finally {
            System.out.print(3);
            throw( new MyExc2() );
        }
    }
    static void n() {
        try {
            System.out.print(4);
            throw( new MyExc3() );
        }
        catch( MyExc1 f ) {
        }
        finally {
            System.out.print(5);
        }
    }
}
```

- A) 1452
- B) 1453Exception in thread "main" MyExc2
- C) 14523Exception in thread "main" MyExc2
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 17 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class A1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            m();
            System.out.print(2);
        }
        catch( MyExc3 b ) {
            System.out.print(3);
        }
        finally {
            System.out.print(4);
            throw( new MyExc3() );
        }
    }
    static void m() {
        try {
            throw( new MyExc1() );
        }
        catch( MyExc1 b ) {
        }
    }
}
```

- A) 124Exception in thread "main" MyExc3
- B) 12
- C) Errore a tempo di compilazione
- D) 1243
- E) Nessuna delle precedenti

### Versione 18 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends Error { }
public class A1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            n();
            System.out.print(2);
        }
        catch( Error t ) {
            throw( new MyExc1() );
        }
        finally {
            throw( new MyExc1() );
        }
    }
    static void n() {
        try {
            throw( new MyExc3() );
        }
        catch( MyExc1 k ) {
            System.out.print(3);
        }
        catch( Error t ) {
            System.out.print(4);
        }
        finally {
            System.out.print(5);
        }
    }
}
```

- A) 152
- B) 15Exception in thread "main" MyExc1
- C) 1452... (ciclo infinito)
- D) Errore a tempo di compilazione
- E) Nessuna delle precedenti

### Versione 19 dell'esercizio 5

Qual è l'output di questo codice?

```
class MyExc1 extends Error { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc2 { }
public class A1 {
    public static void main(String [] argv) {
        try {
            System.out.print(1);
            q();
        }
        catch( MyExc1 x ) {
            System.out.print(2);
            throw( new MyExc2() );
        }
        catch( Error k ) {
            throw( new MyExc2() );
        }
        finally {
            System.out.print(3);
            throw( new Error() );
        }
    }
    static void q() {
        try {
            System.out.print(4);
            throw( new MyExc1() );
        }
        finally {
            System.out.print(5);
        }
    }
}
```

- A) 14523Exception in thread "main" Error
- B) Errore a tempo di compilazione
- C) 145
- D) 145222222... (ciclo infinito)
- E) Nessuna delle precedenti

### Versione 1 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class E {  
    String a;  
    int x;  
    static String g(Object z) {  
        return "in E:" + z;  
    }  
    E() {  
        x = 9;  
        a = g(x);  
    }  
}  
class C extends E {  
    C(String c) {  
        x = 5;  
    }  
    static String g(Object z) {  
        return "in C:" + z;  
    }  
}  
public class Test {  
    public static void main(String [] arg) {  
        C v = new C("abc");  
        System.out.println(v.a + "; " + (int) v.x);  
    }  
}
```

- A) Errore a tempo di compilazione
- B) Stampa in E:9; 5
- C) Stampa in C:9; 5
- D) Stampa null9; 9
- E) Nessuna delle precedenti

## Versione 2 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class G {  
    String s;  
    long v;  
    G(String y) {  
        v = 1;  
        s = m(y);  
    }  
    static String m(Object r) {  
        return "method of G:" + r;  
    }  
}  
class A extends G {  
    static String m(Object r) {  
        return "method of A:" + r;  
    }  
    A() {  
        v = 5;  
    }  
}  
public class Start {  
    public static void main(String [] arg) {  
        G c = new A();  
        System.out.println(c.s + "; " + (int) c.v);  
    }  
}
```

- A) Stampa method of G:1; 5
- B) Stampa method of A:1; 5
- C) Stampa method of A:5; 5
- D) Stampa null1; 1
- E) Nessuna delle precedenti

### Versione 3 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class E {
    String x;
    int k;
    static String p(Object c) {
        return "in E:" + c;
    }
    E() {
        k++;
        x = p(k);
    }
}
class D extends E {
    static String p(Object c) {
        return "in D:" + c;
    }
    D(String d) {
        k++;
    }
}
public class Test {
    public static void main(String [] arg) {
        D y = new D("xxx");
        System.out.println(y.x + "; " + (int) y.k);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa in D:1; 2
- C) Stampa in D:2; 2
- D) Stampa null1; 1
- E) Nessuna delle precedenti

#### Versione 4 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class D {
    String c;
    long z;
    D(String i) {
        z = 1;
        c = g(i);
    }
    String g(Object s) {
        return "D:" + s;
    }
}
class C extends D {
    C() {
        super("xxx");
        z = 4;
    }
    String g(Object s) {
        return "C:" + s;
    }
}
public class Pub {
    public static void main(String [] arg) {
        D b = new C();
        System.out.println(b.c + "; " + (int) b.z);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa D:1; 4
- C) Stampa C:4; 4
- D) Stampa null1; 1
- E) Nessuna delle precedenti

### Versione 5 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class C {
    String c;
    short y;
    C() {
        y++;
        c = m(y);
    }
    String m(Object a) {
        return "in C:" + a;
    }
}
class G extends C {
    G() {
        y++;
    }
    String m(Object a) {
        return "in G:" + a;
    }
}
public class Start {
    public static void main(String [] arg) {
        G n = new G();
        System.out.println(n.c + "; " + (int) n.y);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa in C:1; 2
- C) Stampa in G:1; 2
- D) Stampa null1; 1
- E) Nessuna delle precedenti

## Versione 6 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class B {
    String c;
    short r;
    static String q(Object a) {
        return "in B:" + a;
    }
    B() {
        r++;
        c = q(r);
    }
}
class C extends B {
    C(String i) {
        r++;
    }
    static String q(Object a) {
        return "in C:" + a;
    }
}
public class Start {
    public static void main(String [] arg) {
        C b = new C("xyz");
        System.out.println(b.c + "; " + (int) b.r);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa in B:1; 2
- C) Stampa in C:1; 2
- D) Stampa null1; 1
- E) Nessuna delle precedenti

### Versione 7 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class A {
    String s;
    long k;
    A() {
        k++;
        s = p(k);
    }
    static String p(Object v) {
        return "method of A:" + v;
    }
}
class B extends A {
    static String p(Object v) {
        return "method of B:" + v;
    }
    B() {
        k++;
    }
}
public class Start {
    public static void main(String [] arg) {
        B i = new B();
        System.out.println(i.s + "; " + (int) i.k);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa method of B:1; 2
- C) Stampa method of B:2; 2
- D) Stampa null1; 1
- E) Nessuna delle precedenti

### Versione 8 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class G {  
    String j;  
    int b;  
    G(String v) {  
        b = 5;  
        j = q(v);  
    }  
    String q(Object y) {  
        return "method of G:" + y;  
    }  
}  
class B extends G {  
    B(String v) {  
        b = 1;  
    }  
    String q(Object y) {  
        return "method of B:" + y;  
    }  
}  
public class Pub {  
    public static void main(String [] arg) {  
        B a = new B("xyz");  
        System.out.println(a.j + "; " + (int) a.b);  
    }  
}
```

- A) Errore a tempo di compilazione
- B) Stampa method of G:5; 1
- C) Stampa method of B:5; 1
- D) Stampa null5; 5
- E) Nessuna delle precedenti

### Versione 9 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class E {
    String r;
    short i;
    String h(Object z) {
        return "E:" + z;
    }
    E() {
        i = 5;
        r = h(i);
    }
}
class D extends E {
    String h(Object z) {
        return "D:" + z;
    }
    D() {
        i = 3;
    }
}
public class Pub {
    public static void main(String [] arg) {
        E y = new D();
        System.out.println(y.r + "; " + (int) y.i);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa E:5; 3
- C) Stampa D:3; 3
- D) Stampa null5; 5
- E) Nessuna delle precedenti

### Versione 10 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class G {
    String a;
    double b;
    G() {
        b = 1;
        a = g((int) b);
    }
    String g(Object r) {
        return "in G:" + r;
    }
}
class B extends G {
    B() {
        b = 6;
    }
    String g(Object r) {
        return "in B:" + r;
    }
}
public class Pub {
    public static void main(String [] arg) {
        G z = new B();
        System.out.println(z.a + "; " + (int) z.b);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa in G:1; 6
- C) Stampa in B:1; 6
- D) Stampa null1; 1
- E) Nessuna delle precedenti

### Versione 11 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class A {
    String z;
    double s;
    A() {
        s = 7;
        z = g((int) s);
    }
    static String g(Object y) {
        return "method of A:" + y;
    }
}
class D extends A {
    D() {
        s = 3;
    }
    static String g(Object y) {
        return "method of D:" + y;
    }
}
public class Start {
    public static void main(String [] arg) {
        D c = new D();
        System.out.println(c.z + "; " + (int) c.s);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa method of D:7; 3
- C) Stampa method of D:3; 3
- D) Stampa null7; 7
- E) Nessuna delle precedenti

### Versione 12 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class G {
    String r;
    long v;
    G(String j) {
        v = 5;
        r = q(j);
    }
    String q(Object s) {
        return "G:" + s;
    }
}
class F extends G {
    F(String j) {
        super("abc");
        v = 1;
    }
    String q(Object s) {
        return "F:" + s;
    }
}
public class Test {
    public static void main(String [] arg) {
        F i = new F("abc");
        System.out.println(i.r + "; " + (int) i.v);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa G:5; 1
- C) Stampa F:5; 1
- D) Stampa null5; 5
- E) Nessuna delle precedenti

### Versione 13 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class F {
    String d;
    double s;
    static String g(Object v) {
        return "F:" + v;
    }
    F(String r) {
        s = 4;
        d = g(r);
    }
}
class G extends F {
    static String g(Object v) {
        return "G:" + v;
    }
    G() {
        s = 1;
    }
}
public class Main {
    public static void main(String [] arg) {
        G j = new G();
        System.out.println(j.d + "; " + (int) j.s);
    }
}
```

- A) Stampa F:4; 1
- B) Stampa G:4; 1
- C) Stampa G:1; 1
- D) Stampa null4; 4
- E) Nessuna delle precedenti

### Versione 14 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class E {
    String s;
    long c;
    String f(Object n) {
        return "in E:" + n;
    }
    E(String v) {
        c = 4;
        s = f(v);
    }
}
class B extends E {
    B(String v) {
        c = 3;
    }
    String f(Object n) {
        return "in B:" + n;
    }
}
public class Main {
    public static void main(String [] arg) {
        B k = new B("xxx");
        System.out.println(k.s + "; " + (int) k.c);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa in E:4; 3
- C) Stampa in B:4; 3
- D) Stampa null4; 4
- E) Nessuna delle precedenti

### Versione 15 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class A {
    String i;
    float z;
    String h(Object r) {
        return "method of A:" + r;
    }
    A() {
        z = 7;
        i = h((int) z);
    }
}
class B extends A {
    String h(Object r) {
        return "method of B:" + r;
    }
    B(String k) {
        z = 8;
    }
}
public class Main {
    public static void main(String [] arg) {
        A c = new B("abc");
        System.out.println(c.i + "; " + (int) c.z);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa method of A:7; 8
- C) Stampa method of B:8; 8
- D) Stampa null7; 7
- E) Nessuna delle precedenti

### Versione 16 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class F {
    String y;
    long r;
    static String f(Object k) {
        return "in F:" + k;
    }
    F() {
        r++;
        y = f(r);
    }
}
class C extends F {
    C(String b) {
        r++;
    }
    static String f(Object k) {
        return "in C:" + k;
    }
}
public class Start {
    public static void main(String [] arg) {
        F z = new C("xxx");
        System.out.println(z.y + "; " + (int) z.r);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa in F:1; 2
- C) Stampa in C:1; 2
- D) Stampa null1; 1
- E) Nessuna delle precedenti

### Versione 17 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class G {
    String z;
    double j;
    String h(Object k) {
        return "method of G:" + k;
    }
    G(String r) {
        j++;
        z = h(r);
    }
}
class D extends G {
    D(String r) {
        super("xxx");
        j++;
    }
    String h(Object k) {
        return "method of D:" + k;
    }
}
public class Main {
    public static void main(String [] arg) {
        D v = new D("xxx");
        System.out.println(v.z + "; " + (int) v.j);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa method of G:1; 2
- C) Stampa method of D:2; 2
- D) Stampa null1; 1
- E) Nessuna delle precedenti

### Versione 18 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class F {
    String v;
    long y;
    F() {
        y = 6;
        v = h(y);
    }
    String h(Object z) {
        return "method of F:" + z;
    }
}
class E extends F {
    String h(Object z) {
        return "method of E:" + z;
    }
    E(String a) {
        y = 2;
    }
}
public class Pub {
    public static void main(String [] arg) {
        E i = new E("xyz");
        System.out.println(i.v + "; " + (int) i.y);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa method of F:6; 2
- C) Stampa method of E:6; 2
- D) Stampa null6; 6
- E) Nessuna delle precedenti

### Versione 19 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class A {
    String k;
    short z;
    String q(Object c) {
        return "method of A:" + c;
    }
    A(String a) {
        z++;
        k = q(a);
    }
}
class B extends A {
    String q(Object c) {
        return "method of B:" + c;
    }
    B(String a) {
        super("abc");
        z++;
    }
}
public class Start {
    public static void main(String [] arg) {
        A i = new B("abc");
        System.out.println(i.k + "; " + (int) i.z);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa method of A:1; 2
- C) Stampa method of B:2; 2
- D) Stampa null1; 1
- E) Nessuna delle precedenti

### Versione 1 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class E {
    String x;
    short a;
    E() {
        a++;
        x = h(a);
    }
    String h(Object y) {
        return "in E:" + y;
    }
}
class G extends E {
    G() {
        a++;
    }
    String h(Object y) {
        return "in G:" + y;
    }
}
public class Test {
    public static void main(String [] arg) {
        G n = new G();
        System.out.println(n.x + "; " + (int) n.a);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa in E:1; 2
- C) Stampa in G:2; 2
- D) Stampa null1; 1
- E) Nessuna delle precedenti

## Versione 2 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class C {
    String k;
    double x;
    static String q(Object n) {
        return "in C:" + n;
    }
    C(String r) {
        x++;
        k = q(r);
    }
}
class E extends C {
    static String q(Object n) {
        return "in E:" + n;
    }
    E(String r) {
        super("xxx");
        x++;
    }
}
public class Pub {
    public static void main(String [] arg) {
        E s = new E("xxx");
        System.out.println(s.k + "; " + (int) s.x);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa in C:xxx; 2
- C) Stampa in E:xxx; 2
- D) Stampa null1; 1
- E) Nessuna delle precedenti

### Versione 3 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class G {
    String n;
    int r;
    String q(Object z) {
        return "method of G:" + z;
    }
    G(String d) {
        r++;
        n = q(d);
    }
}
class B extends G {
    B(String d) {
        super("abc");
        r++;
    }
    String q(Object z) {
        return "method of B:" + z;
    }
}
public class Pub {
    public static void main(String [] arg) {
        B j = new B("abc");
        System.out.println(j.n + "; " + (int) j.r);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa method of G:abc; 2
- C) Stampa method of B:abc; 1
- D) Stampa null1; 1
- E) Nessuna delle precedenti

#### Versione 4 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class A {  
    String n;  
    int j;  
    A(String k) {  
        j = 2;  
        n = f(k);  
    }  
    String f(Object a) {  
        return "in A:" + a;  
    }  
}  
class D extends A {  
    String f(Object a) {  
        return "in D:" + a;  
    }  
    D() {  
        j = 9;  
    }  
}  
public class Pub {  
    public static void main(String [] arg) {  
        D r = new D();  
        System.out.println(r.n + "; " + (int) r.j);  
    }  
}
```

- A) Stampa in A:2; 9
- B) Stampa in D:2; 9
- C) Stampa in D:9; 9
- D) Stampa null2; 2
- E) Nessuna delle precedenti

### Versione 5 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class B {
    String x;
    long c;
    String q(Object j) {
        return "method of B:" + j;
    }
    B() {
        c = 2;
        x = q(c);
    }
}
class F extends B {
    String q(Object j) {
        return "method of F:" + j;
    }
    F() {
        c = 9;
    }
}
public class Main {
    public static void main(String [] arg) {
        F z = new F();
        System.out.println(z.x + "; " + (int) z.c);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa method of B; 9
- C) Stampa method of F; 9
- D) Stampa null; 2
- E) Nessuna delle precedenti

## Versione 6 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class B {
    String s;
    short n;
    B() {
        n = 2;
        s = p(n);
    }
    String p(Object c) {
        return "in B:" + c;
    }
}
class G extends B {
    G(String y) {
        n = 9;
    }
    String p(Object c) {
        return "in G:" + c;
    }
}
public class Main {
    public static void main(String [] arg) {
        G v = new G("abc");
        System.out.println(v.s + "; " + (int) v.n);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa in B:2; 9
- C) Stampa in G:9; 9
- D) Stampa null2; 2
- E) Nessuna delle precedenti

### Versione 7 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class E {
    String s;
    int c;
    E() {
        c = 2;
        s = m(c);
    }
    String m(Object n) {
        return "E:" + n;
    }
}
class G extends E {
    G() {
        c = 3;
    }
    String m(Object n) {
        return "G:" + n;
    }
}
public class Test {
    public static void main(String [] arg) {
        E j = new G();
        System.out.println(j.s + "; " + (int) j.c);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa E:2; 3
- C) Stampa G:3; 3
- D) Stampa null2; 2
- E) Nessuna delle precedenti

### Versione 8 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class C {  
    String k;  
    short y;  
    C(String z) {  
        y = 5;  
        k = h(z);  
    }  
    static String h(Object v) {  
        return "in C:" + v;  
    }  
}  
class E extends C {  
    E() {  
        y = 8;  
    }  
    static String h(Object v) {  
        return "in E:" + v;  
    }  
}  
public class Start {  
    public static void main(String [] arg) {  
        E x = new E();  
        System.out.println(x.k + "; " + (int) x.y);  
    }  
}
```

- A) Stampa in C:5; 8
- B) Stampa in E:5; 8
- C) Stampa in E:8; 8
- D) Stampa null5; 5
- E) Nessuna delle precedenti

### Versione 9 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class G {
    String z;
    int v;
    G() {
        v++;
        z = m(v);
    }
    static String m(Object d) {
        return "in G:" + d;
    }
}
class E extends G {
    static String m(Object d) {
        return "in E:" + d;
    }
    E() {
        v++;
    }
}
public class Pub {
    public static void main(String [] arg) {
        G i = new E();
        System.out.println(i.z + "; " + (int) i.v);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa in G:1; 2
- C) Stampa in E:1; 2
- D) Stampa null1; 1
- E) Nessuna delle precedenti

### Versione 10 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class F {
    String n;
    double j;
    F() {
        j++;
        n = g((int) j);
    }
    static String g(Object b) {
        return "F:" + b;
    }
}
class C extends F {
    C() {
        j++;
    }
    static String g(Object b) {
        return "C:" + b;
    }
}
public class Main {
    public static void main(String [] arg) {
        C c = new C();
        System.out.println(c.n + "; " + (int) c.j);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa C:1; 2
- C) Stampa C:2; 2
- D) Stampa null1; 1
- E) Nessuna delle precedenti

### Versione 11 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class C {
    String x;
    int i;
    C(String k) {
        i++;
        x = p(k);
    }
    static String p(Object d) {
        return "method of C:" + d;
    }
}
class A extends C {
    A(String k) {
        super("xyz");
        i++;
    }
    static String p(Object d) {
        return "method of A:" + d;
    }
}
public class Main {
    public static void main(String [] arg) {
        C y = new A("xyz");
        System.out.println(y.x + "; " + (int) y.i);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa method of C:xyz; 2
- C) Stampa method of A:xyz; 2
- D) Stampa null1; 1
- E) Nessuna delle precedenti

### Versione 12 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class F {
    String x;
    int d;
    String q(Object a) {
        return "method of F:" + a;
    }
    F(String j) {
        d = 9;
        x = q(j);
    }
}
class D extends F {
    D(String j) {
        super("abc");
        d = 3;
    }
    String q(Object a) {
        return "method of D:" + a;
    }
}
public class Main {
    public static void main(String [] arg) {
        F r = new D("abc");
        System.out.println(r.x + "; " + (int) r.d);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa method of F:abc; 3
- C) Stampa method of D:abc; 3
- D) Stampa null9; 9
- E) Nessuna delle precedenti

### Versione 13 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class C {
    String r;
    double n;
    C(String x) {
        n = 4;
        r = h(x);
    }
    String h(Object z) {
        return "C:" + z;
    }
}
class E extends C {
    E() {
        n = 8;
    }
    String h(Object z) {
        return "E:" + z;
    }
}
public class Start {
    public static void main(String [] arg) {
        C c = new E();
        System.out.println(c.r + "; " + (int) c.n);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa C:4; 8
- C) Stampa E:4; 8
- D) Stampa null4; 4
- E) Nessuna delle precedenti

### Versione 14 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class D {  
    String v;  
    float c;  
    D() {  
        c = 4;  
        v = g((int) c);  
    }  
    String g(Object k) {  
        return "in D:" + k;  
    }  
}  
class F extends D {  
    F(String z) {  
        c = 6;  
    }  
    String g(Object k) {  
        return "in F:" + k;  
    }  
}  
public class Test {  
    public static void main(String [] arg) {  
        D b = new F("abc");  
        System.out.println(b.v + "; " + (int) b.c);  
    }  
}
```

- A) Errore a tempo di compilazione
- B) Stampa in D:abc; 6
- C) Stampa in F:abc; 6
- D) Stampa nullabc; 4
- E) Nessuna delle precedenti

### Versione 15 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class C {
    String k;
    long n;
    String m(Object s) {
        return "method of C:" + s;
    }
    C() {
        n++;
        k = m(n);
    }
}
class G extends C {
    G() {
        n++;
    }
    String m(Object s) {
        return "method of G:" + s;
    }
}
public class Test {
    public static void main(String [] arg) {
        G y = new G();
        System.out.println(y.k + "; " + (int) y.n);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa method of C:1; 2
- C) Stampa method of G:2; 2
- D) Stampa null1; 1
- E) Nessuna delle precedenti

### Versione 16 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class C {
    String y;
    float s;
    C() {
        s++;
        y = p((int) s);
    }
    String p(Object v) {
        return "in C:" + v;
    }
}
class D extends C {
    D(String i) {
        s++;
    }
    String p(Object v) {
        return "in D:" + v;
    }
}
public class Pub {
    public static void main(String [] arg) {
        C n = new D("xxx");
        System.out.println(n.y + "; " + (int) n.s);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa in C1; 2
- C) Stampa in D:1; 2
- D) Stampa null1; 1
- E) Nessuna delle precedenti

### Versione 17 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class F {
    String x;
    float r;
    String h(Object i) {
        return "method of F:" + i;
    }
    F(String n) {
        r++;
        x = h(n);
    }
}
class B extends F {
    B(String n) {
        r++;
    }
    String h(Object i) {
        return "method of B:" + i;
    }
}
public class Main {
    public static void main(String [] arg) {
        F k = new B("xyz");
        System.out.println(k.x + "; " + (int) k.r);
    }
}
```

- A) Stampa method of F:xyz; 2
- B) Stampa method of B:xyz; 2
- C) Stampa method of B:xyz; 2
- D) Stampa nullxyz; 1
- E) Nessuna delle precedenti

### Versione 18 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class B {
    String z;
    double n;
    String f(Object x) {
        return "in B:" + x;
    }
    B() {
        n = 3;
        z = f((int) n);
    }
}
class G extends B {
    G(String v) {
        n = 7;
    }
    String f(Object x) {
        return "in G:" + x;
    }
}
public class Pub {
    public static void main(String [] arg) {
        B r = new G("xyz");
        System.out.println(r.z + "; " + (int) r.n);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa in B:3; 7
- C) Stampa in G:7; 7
- D) Stampa null3; 3
- E) Nessuna delle precedenti

### Versione 19 dell'esercizio 6

Qual è il comportamento del seguente programma?

```
class F {
    String v;
    double d;
    String m(Object i) {
        return "in F:" + i;
    }
    F(String x) {
        d++;
        v = m(x);
    }
}
class G extends F {
    String m(Object i) {
        return "in G:" + i;
    }
    G(String x) {
        d++;
    }
}
public class Test {
    public static void main(String [] arg) {
        F j = new G("abc");
        System.out.println(j.v + "; " + (int) j.d);
    }
}
```

- A) Errore a tempo di compilazione
- B) Stampa in F:1; 2
- C) Stampa in G:1; 2
- D) Stampa null1; 1
- E) Nessuna delle precedenti

## Versione 1 dell'esercizio 6

Qual è l'output di questo codice?

```
class Animale {  
  
    String famiglia;  
    protected Animale (String f){  
        famiglia = f;  
    }  
  
    protected void stampa() {  
        System.out.println("Animale: "  
+ famiglia);  
    }  
}  
  
class Mammifero extends Animale {  
    public final Mammifero(String f){  
        super(f);  
    }  
  
    public void stampa() {  
        System.out.println("Mammifero: "  
+ famiglia);  
    }  
    public static void main(String[] args){  
        Animale a = new Mammifero("Canidi");  
        a.stampa();  
    }  
}
```

- A) Animale: Canidi
- B) Mammifero: Canidi
- C) Errore a tempo di compilazione
- D) Errore a tempo di esecuzione
- E) Nessuna delle precedenti

## Versione 2 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente metodo statico?

```
class C {  
    int x;  
    public static void main(String[] args){  
        boolean y;  
        System.out.println("x=" + x); //1  
        System.out.println("y=" + y); //2  
    }  
}
```

- A) Errore di compilazione a linea 1.
- B) Errore di compilazione a linea 2.
- C) Errore di compilazione a linea 1 e linea 2.
- D) Errore di esecuzione.
- E) Nessuna delle precedenti.

### Versione 3 dell'esercizio 6

Qual è l'output di questo codice?

```
abstract class Programmatore {  
    String nome;  
    String cognome;  
    int anniEsperienza;  
  
    public Programmatore(String n, String c, int a){  
        nome=n;  
        cognome=c;  
        anniEsperienza=a;  
    }  
    abstract void programma();  
}  
public class ProgrammatoreJava extends Programmatore {  
  
    void stampa(){  
  
        System.out.print("Programmatore Java con ");  
        System.out.print(anniEsperienza);  
        System.out.println(" anni di esperienza");  
    }  
  
    ProgrammatoreJava(String n, String c, int a){  
super(n,c,a);  
    }  
    public static void main(String [] argv) {  
        ProgrammatoreJava pj=new  
            ProgrammatoreJava("Franco", "Bianchi",5);  
        pj.stampa();  
    }  
}
```

- A) Errore a tempo di compilazione
- B) Errore a tempo di esecuzione
- C) Programmatore Java con 5 anni di esperienza
- D) Programmatore Java con 0 anni di esperienza
- E) Nessuna delle precedenti

#### Versione 4 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
interface A {
    String name = "A";
    void m();
}

abstract class B implements A {
    String name = "B";
    abstract B g();
}

class C extends B {
    String name = "C";
    void m() {
        System.out.print(
            ((A)this).name +
            ((B)this).name +
            ((C)this).name);
    }
    B g() {return this;}
    public static void main(String[] args){
        new C().m();
    }
}
```

- A) ABC
- B) CCC
- C) Errore a tempo di compilazione.
- D) Errore a tempo di esecuzione.
- E) Risultato diverso dai precedenti.

### Versione 5 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public abstract class X {  
    X(){  
        System.out.print("X");  
    }  
    abstract void z();  
}  
  
class Y extends X {  
    void z(){  
        Y(){  
            System.out.print("Y");  
        }  
    }  
    public static void main(String[] args){  
        X x = new Y();  
    }  
}
```

- A) X
- B) Y
- C) Errore a tempo di compilazione.
- D) Errore a tempo di esecuzione.
- E) Nessuna delle precedenti.

## Versione 6 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public abstract class A {  
    A(){  
        System.out.print("A");  
    }  
    abstract void f();  
}  
  
class B extends A {  
    void f(){}
    B(){  
        System.out.print("B");
    }
}  
  
public static void main(String[] args) {
    A a = new B();
}
}
```

- A) Errore a tempo di compilazione.
- B) Errore a tempo di esecuzione.
- C) Stampa: A
- D) Stampa: B
- E) Nessuna delle precedenti.

### **Versione 7 dell'esercizio 6**

Quale delle seguenti è un sovraccaricamento illegale del metodo m nella classe C?

```
public class C {  
    final void m(int x, Object y) {}  
}
```

- A) public String m(int x, String y)  
{return x + y;}
- B) public int m(int x, Object o)  
{return x;}
- C) private void m(String y) {}
- D) public int m(int x){return x;}
- E) Nessuna delle precedenti.

### Versione 8 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public abstract class A {  
    A(){  
        System.out.print("A");  
    }  
    abstract void f();  
}  
  
class B extends A {  
    void f(){}
    B(){  
        System.out.print("B");
    }
}  
  
public static void main(String[] args) {
    A a = new B();
}
}
```

- A) Errore a tempo di compilazione.
- B) Errore a tempo di esecuzione.
- C) Stampa: A
- D) Stampa: B
- E) Nessuna delle precedenti.

### Versione 9 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public abstract class A {  
    A(int i){  
        System.out.print("A");  
    }  
    abstract void f();  
}  
  
class B extends A {  
    void f(){  
        B(){  
            System.out.print("B");  
        }  
    }  
    public static void main(String[] args) {  
        A a = new B();  
    }  
}
```

- A) Errore a tempo di compilazione.
- B) Errore a tempo di esecuzione.
- C) Stampa: A
- D) Stampa: B
- E) Nessuna delle precedenti.

### Versione 10 dell'esercizio 6

Cosa succede compilando e, se possibile, eseguendo questo codice

```
class C {  
    int x;  
    C(){ x=1; }  
    public void m() {  
        System.out.println("x di C vale "+x);  
    }  
}  
  
class D extends C {  
    int x = 2;  
    public void m() {  
        System.out.println("x di D vale "+x);  
    }  
}  
  
public class Test {  
    int x = 3;  
    public static void main(String[] a) {  
        C y = new D();  
        y.m();  
    }  
}
```

- A) Errore a tempo di compilazione
- B) Stampa x di C vale 1
- C) Stampa x di C vale 2
- D) Stampa x di D vale 1
- E) Stampa x di D vale 2

### Versione 11 dell'esercizio 6

Qual è l'output di questo codice?

```
interface FiguraGeometrica {  
    static int dimensioni = 10;  
    void print();  
}  
  
class Cono implements FiguraGeometrica {  
  
    static final int numLati = 3;  
  
    public void print() {  
        System.out.println("Cono: "  
            + dimensioni + " - " +  
            numLati + " lati per faccia.");  
    }  
  
    public static void main(String [] argv) {  
        Cono c1 = new Cono();  
        FiguraGeometrica f1;  
        f1= (FiguraGeometrica) f1;  
        c1.print();  
    }  
}
```

- A) Errore a tempo di compilazione
- B) Errore a tempo di esecuzione
- C) Cono: 8 - 3 lati per faccia
- D) Cono: 3 - 3 lati per faccia
- E) Nessuna delle precedenti

### Versione 12 dell'esercizio 6

Qual è l'output di questo codice?

```
interface FiguraGeometrica {  
    static int dimensioni = 10;  
    void print();  
}  
  
class Cono implements FiguraGeometrica {  
  
    static final int numLati = 3;  
  
    public void print() {  
        System.out.println("Cono: "  
            + dimensioni + " - " +  
            numLati + " lati per faccia.");  
    }  
  
    public static void main(String [] argv) {  
        Cono c1 = new Cono();  
        FiguraGeometrica f1 =c1;  
        c1.print();  
    }  
}
```

- A) Errore a tempo di compilazione
- B) Errore a tempo di esecuzione
- C) Cono: 8 - 3 lati per faccia
- D) Cono: 3 - 3 lati per faccia
- E) Nessuna delle precedenti

### Versione 13 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public class Person {
    String name, surname;

    Person(String n, String s) {
        name = n;
        surname = s;
    }

    public String id() {
        return name + " ";
    }
}

class Employee extends Person {
    Employee(String n, String s) {super(n,s);}

    public String id() {
        return name + " " + surname + " ";
    }
}

class Main {
    public static void main(String[] args) {
        Person a = new Person("Ann","Taylor");
        Person e = new Employee("Eva","Smith");
        Employee j = new Employee("Jim","Brown");
        System.out.print(a.id());
        System.out.print(e.id());
        System.out.print(j.id());
    }
}
```

- A) Ann Eva Jim Brown
- B) Ann Eva Smith Jim Brown
- C) Jim Jim Jim Brown
- D) Errore di compilazione.
- E) Errore di esecuzione.

### **Versione 14 dell'esercizio 6**

Date le seguenti classi:

```
class A {  
    Integer i = 6;  
}  
class B extends A {  
}
```

Quale delle seguenti proposte di cambiamento del codice precedente è la migliore rispetto all'incapsulamento?

- A) Le classi sono perfettamente incapsulate.
- B) Definire **private** la variabile i e aggiungere due metodi, uno che ne modifichi ed uno che ne restituisca il valore.
- C) Definire **final** la variabile i.
- D) Definire **protected** la variabile i e aggiungere due metodi, uno che ne modifichi ed uno che ne restituisca il valore.
- E) Nessuna delle precedenti.

### Versione 15 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public abstract class X {  
    X(){  
        System.out.print("X");  
    }  
    abstract void z();  
}  
  
class Y extends X {  
    void z(){  
        Y(){  
            System.out.print("Y");  
        }  
    }  
    public static void main(String[] args){  
        X x = new Y();  
    }  
}
```

- A) X
- B) Y
- C) Errore a tempo di compilazione.
- D) Errore a tempo di esecuzione.
- E) Nessuna delle precedenti.

### Versione 16 dell'esercizio 6

Cosa succede compilando e, se possibile, eseguendo questo codice

```
class C {  
    static int x;  
    static { x=1; }  
    public void m() {  
        System.out.println("x di C vale "+x);  
    }  
}  
  
class D extends C {  
    int x = 2;  
    public void m() {  
        System.out.println("x di D vale "+x);  
    }  
}  
  
public class Test {  
    int x = 3;  
    public static void main(String[] a) {  
        C y = new D();  
        y.m();  
    }  
}
```

- A) Errore a tempo di compilazione
- B) Stampa x di C vale 1
- C) Stampa x di C vale 2
- D) Stampa x di D vale 1
- E) Stampa x di D vale 2

### Versione 17 dell'esercizio 6

Qual è l'output di questo codice?

```
class Animale {  
  
    String famiglia;  
    protected Animale(){}
    protected Animale (String f){  
        famiglia = f;  
    }  
  
    protected void stampa() {  
        System.out.println("Animale: "  
+ famiglia);  
    }  
}  
  
class Mammifero extends Animale {  
    public Mammifero(String f){}  
  
    public void stampa() {  
        System.out.println("Mammifero: "  
+ famiglia);  
    }  
    public static void main(String[] args){  
        Animale a = new Mammifero("Canidi");  
        a.stampa();  
    }  
}
```

- A) Animale: Canidi
- B) Mammifero: Canidi
- C) Errore a tempo di compilazione
- D) Errore a tempo di esecuzione
- E) Nessuna delle precedenti

### Versione 18 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public class Person {
    static String name;
    static int age;

    Person(String n, int a) {
        name = n;
        age = a;
    }

    public static String print() {
        return name + " ";
    }
}

class Student extends Person {
    Student(String n, int a) {super(n,a);}

    public static String print() {
        return name + " " + age + " ";
    }
}

class Test {
    public static void main(String[] args){
        Person e = new Student("A S", 20);
        Student j = new Student("B J", 25);
        System.out.print(e.print());
        System.out.print(j.print());
    }
}
```

- A) A S B J 25
- B) A S 20 B J 25
- C) Errore a tempo di compilazione.
- D) Errore a tempo di esecuzione.
- E) Nessuna delle precedenti.

### Versione 19 dell'esercizio 6

Cosa succede compilando e, se possibile, eseguendo questo codice

```
class C {  
    static int x;  
    static { x=1; }  
    public void m() {  
        System.out.println("x di C vale "+x);  
    }  
}  
  
class D extends C {  
    int x = 2;  
    public void m() {  
        System.out.println("x di D vale "+x);  
    }  
}  
  
public class Test {  
    int x = 3;  
    public static void main(String[] a) {  
        C y = new D();  
        y.m();  
    }  
}
```

- A) Errore a tempo di compilazione
- B) Stampa x di C vale 1
- C) Stampa x di C vale 2
- D) Stampa x di D vale 1
- E) Stampa x di D vale 2

### Versione 20 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public class Person {
    String name;
    int age;

    Person(String n, int a) {
        name = n;
        age = a;
    }

    public String print() {
        return name + " ";
    }
}

class Student extends Person {
    Student(String n, int a) {super(n,a);}

    public String print() {
        return name + " " + age + " ";
    }
}

class Test {
    public static void main(String[] args){
        Person e = new Student("A S", 20);
        Student j = new Student("B J", 25);
        System.out.print(e.print());
        System.out.print(j.print());
    }
}

A) A S B J 25
B) A S 20 B J 25
C) Errore a tempo di compilazione.
D) Errore a tempo di esecuzione.
E) Nessuna delle precedenti.
```

### Versione 21 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
interface A {
    String name = "A";
    void m();
}

abstract class B implements A {
    String name = "B";
    abstract B g();
}

class C extends B {
    String name = "C";
    void m() {
        System.out.print(
            ((A)this).name +
            ((B)this).name +
            ((C)this).name);
    }
    B g() {return this;}
    public static void main(String[] args){
        new C().m();
    }
}
```

- A) ABC
- B) CCC
- C) Errore a tempo di compilazione.
- D) Errore a tempo di esecuzione.
- E) Risultato diverso dai precedenti.

## Versione 22 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
interface A {
    String name = "A";
    void m();
}

abstract class B implements A {
    String name = "B";
    abstract B g();
}

class C extends B {
    String name = "C";
    public void m() {
        System.out.print(
            ((A)this).name +
            ((B)this).name +
            ((C)this).name);
    }
    public B g() {return this;}
    public static void main(String[] args){
        new C().m();
    }
}
```

- A) ABC
- B) CCC
- C) Errore a tempo di compilazione.
- D) Errore a tempo di esecuzione.
- E) Risultato diverso dai precedenti.

### Versione 23 dell'esercizio 6

Cosa succede compilando e, se possibile, eseguendo questo codice

```
class C {  
    static int x;  
    C(){ x=1; }  
    public static void m() {  
        System.out.println("x di C vale "+x);  
    }  
}  
  
class D extends C {  
    static int x = 2;  
    public static void m() {  
        System.out.println("x di D vale "+x);  
    }  
}  
  
public class Test {  
    int x = 3;  
    public static void main(String[] a) {  
        C y = new D();  
        y.m();  
    }  
}
```

- A) Errore a tempo di compilazione
- B) Stampa x di C vale 1
- C) Stampa x di C vale 2
- D) Stampa x di D vale 1
- E) Stampa x di D vale 2

### Versione 24 dell'esercizio 6

Qual è l'output di questo codice?

```
abstract class Programmatore {  
    String nome;  
    String cognome;  
    int anniEsperienza;  
  
    public Programmatore(String n, String c, int a){  
        nome=n;  
        cognome=c;  
        anniEsperienza=a;  
    }  
    abstract void programma();  
}  
public class ProgrammatoreJava extends Programmatore {  
  
    void stampa(){  
  
        System.out.print("Programmatore Java con ");  
        System.out.print(anniEsperienza);  
        System.out.println(" anni di esperienza");  
    }  
  
    ProgrammatoreJava(String n, String c, int a){  
super(n,c,a);  
    }  
    public static void main(String [] argv) {  
        ProgrammatoreJava pj=new  
            ProgrammatoreJava("Franco", "Bianchi",5);  
        pj.stampa();  
    }  
}
```

- A) Errore a tempo di compilazione
- B) Errore a tempo di esecuzione
- C) Programmatore Java con 5 anni di esperienza
- D) Programmatore Java con 0 anni di esperienza
- E) Nessuna delle precedenti

### Versione 25 dell'esercizio 6

Qual è l'output di questo codice?

```
class Animale {  
  
    String famiglia;  
    protected Animale (String f){  
        famiglia = f;  
    }  
  
    protected void stampa() {  
        System.out.println("Animale: "  
+ famiglia);  
    }  
}  
  
class Mammifero extends Animale {  
    public final Mammifero(String f){  
        super(f);  
    }  
  
    public void stampa() {  
        System.out.println("Mammifero: "  
+ famiglia);  
    }  
    public static void main(String[] args){  
        Animale a = new Mammifero("Canidi");  
        a.stampa();  
    }  
}
```

- A) Animale: Canidi
- B) Mammifero: Canidi
- C) Errore a tempo di compilazione
- D) Errore a tempo di esecuzione
- E) Nessuna delle precedenti

### Versione 26 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
interface A {
    String name = "A";
    void m();
}

abstract class B implements A {
    String name = "B";
    abstract B g();
}

class C extends B {
    String name = "C";
    public void m() {
        System.out.print(
            ((A)this).name +
            ((B)this).name +
            ((C)this).name);
    }
    public B g() {return this;}
    public static void main(String[] args){
        new C().m();
    }
}
```

- A) ABC
- B) CCC
- C) Errore a tempo di compilazione.
- D) Errore a tempo di esecuzione.
- E) Risultato diverso dai precedenti.

### Versione 27 dell'esercizio 6

Qual è l'output di questo codice?

```
1. public class C {  
2.     static void f(int i) {  
3.         System.out.println("int " + i);  
4.     }  
5.     static void f(float f) {  
6.         System.out.println("float "+ f);  
7.     }  
8.     public static void main(String[] args){  
9.         long l = 1;  
10.        double d = 5.0f;  
11.        f(l);  
12.        f(d);  
13.    }  
14. }
```

- A) Stampa int 1 float 5.0
- B) Stampa float 1.0 float 5.0
- C) Errore a tempo di compilazione a linea 11.
- D) Errore a tempo di compilazione a linea 12.
- E) Errore a tempo di esecuzione

### **Versione 28 dell'esercizio 6**

Dato un file contenente il seguente codice:

```
package pk;

public class C {
    public static class I {
    }
}
```

Dire quale delle seguenti affermazioni è vera:

- A) Un'istanza della classe I può essere costruita solo all'interno della classe C.
- B) Un'istanza della classe I può essere costruita solo all'interno del pacchetto pk.
- C) All'interno del pacchetto pk, un'istanza della classe I può essere costruita con `new I()`.
- D) Un'istanza della classe I può essere costruita con `new C.I()`.
- E) Nessuna delle precedenti.

### Versione 29 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public abstract class A {  
    A(){  
        System.out.print("A");  
    }  
    abstract void f();  
}  
  
class B extends A {  
    void f(){}
    B(){  
        System.out.print("B");
    }
}  
  
public static void main(String[] args) {
    A a = new B();
}
}
```

- A) Errore a tempo di compilazione.
- B) Errore a tempo di esecuzione.
- C) Stampa: A
- D) Stampa: B
- E) Nessuna delle precedenti.

### Versione 1 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public abstract class X {  
    X(long l){  
        System.out.print("X");  
    }  
    abstract void z();  
}  
  
class Y extends X {  
    void z(){  
        Y(){  
            System.out.print("Y");  
        }  
    }  
  
    public static void main(String[] args){  
        X x = new Y();  
    }  
}
```

- A) X
- B) Y
- C) Errore a tempo di compilazione.
- D) Errore a tempo di esecuzione.
- E) Nessuna delle precedenti.

## Versione 2 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public abstract class X {  
    X(){  
        System.out.print("X");  
    }  
    abstract void z();  
}  
  
class Y extends X {  
    void z(){  
        Y(){  
            System.out.print("Y");  
        }  
    }  
    public static void main(String[] args){  
        X x = new Y();  
    }  
}
```

- A) X
- B) Y
- C) Errore a tempo di compilazione.
- D) Errore a tempo di esecuzione.
- E) Nessuna delle precedenti.

### Versione 3 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public abstract class X {  
    X(long l){  
        System.out.print("X");  
    }  
    abstract void z();  
}  
  
class Y extends X {  
    void z(){  
        Y(){  
            System.out.print("Y");  
        }  
    }  
  
    public static void main(String[] args){  
        X x = new Y();  
    }  
}
```

- A) X
- B) Y
- C) Errore a tempo di compilazione.
- D) Errore a tempo di esecuzione.
- E) Nessuna delle precedenti.

#### Versione 4 dell'esercizio 6

Cosa succede compilando e, se possibile, eseguendo questo codice

```
class C {  
    static int x;  
    static { x=1; }  
    public void m() {  
        System.out.println("x di C vale "+x);  
    }  
}  
  
class D extends C {  
    int x = 2;  
    public void m() {  
        System.out.println("x di D vale "+x);  
    }  
}  
  
public class Test {  
    int x = 3;  
    public static void main(String[] a) {  
        C y = new D();  
        y.m();  
    }  
}
```

- A) Errore a tempo di compilazione
- B) Stampa x di C vale 1
- C) Stampa x di C vale 2
- D) Stampa x di D vale 1
- E) Stampa x di D vale 2

### **Versione 5 dell'esercizio 6**

Dato un file contenente il seguente codice:

```
package pk;

public class A {
    public final class I {
    }
}
```

Dire quale delle seguenti affermazioni è vera:

- A) Un'istanza della classe I può essere costruita solo all'interno della classe A.
- B) Un'istanza della classe I può essere costruita solo all'interno del pacchetto pk.
- C) Un'istanza della classe I può essere costruita con `sf new A.I()`.
- D) Un'istanza della classe I può essere costruita con `new A().new I()`.
- E) Nessuna delle precedenti.

## Versione 6 dell'esercizio 6

Qual è l'output di questo codice?

```
1. public class C {  
2.     static void f(int i) {  
3.         System.out.println("int " + i);  
4.     }  
5.     static void f(float f) {  
6.         System.out.println("float "+ f);  
7.     }  
8.     public static void main(String[] args){  
9.         long l = 1;  
10.        double d = 5.0f;  
11.        f(l);  
12.        f(d);  
13.    }  
14. }
```

- A) Stampa int 1 float 5.0
- B) Stampa float 1.0 float 5.0
- C) Errore a tempo di compilazione a linea 11.
- D) Errore a tempo di compilazione a linea 12.
- E) Errore a tempo di esecuzione

### Versione 7 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public class Person {
    String name, surname;

    Person(String n, String s) {
        name = n;
        surname = s;
    }

    public String id() {
        return name + " ";
    }
}

class Employee extends Person {
    Employee(String n, String s) {super(n,s);}

    public String id() {
        return name + " " + surname + " ";
    }
}

class Main {
    public static void main(String[] args) {
        Person a = new Person("Ann","Taylor");
        Person e = new Employee("Eva","Smith");
        Employee j = new Employee("Jim","Brown");
        System.out.print(a.id());
        System.out.print(e.id());
        System.out.print(j.id());
    }
}
```

- A) Ann Eva Jim Brown
- B) Ann Eva Smith Jim Brown
- C) Jim Jim Jim Brown
- D) Errore di compilazione.
- E) Errore di esecuzione.

### Versione 8 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
interface A {
    String name = "A";
    void m();
}

abstract class B implements A {
    String name = "B";
    abstract B g();
}

class C extends B {
    String name = "C";
    void m() {
        System.out.print(
            ((A)this).name +
            ((B)this).name +
            ((C)this).name);
    }
    B g() {return this;}
    public static void main(String[] args){
        new C().m();
    }
}
```

- A) ABC
- B) CCC
- C) Errore a tempo di compilazione.
- D) Errore a tempo di esecuzione.
- E) Risultato diverso dai precedenti.

### Versione 9 dell'esercizio 6

Quale delle seguenti è un sovraccaricamento illegale del metodo m nella classe C?

```
public class C {  
    final void m(int x, Object y) {}  
}
```

- A) public String m(int x, String y)  
{return x + y;}
- B) public int m(int x, Object o)  
{return x;}
- C) private void m(String y) {}
- D) public int m(int x){return x;}
- E) Nessuna delle precedenti.

### Versione 10 dell'esercizio 6

Qual è l'output di questo codice?

```
class Animale {  
  
    String famiglia;  
    protected Animale (String f){  
        famiglia = f;  
    }  
  
    protected void stampa() {  
        System.out.println("Animale: "  
+ famiglia);  
    }  
}  
  
class Mammifero extends Animale {  
    public final Mammifero(String f){  
        super(f);  
    }  
  
    public void stampa() {  
        System.out.println("Mammifero: "  
+ famiglia);  
    }  
    public static void main(String[] args){  
        Animale a = new Mammifero("Canidi");  
        a.stampa();  
    }  
}
```

- A) Animale: Canidi
- B) Mammifero: Canidi
- C) Errore a tempo di compilazione
- D) Errore a tempo di esecuzione
- E) Nessuna delle precedenti

### Versione 11 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public abstract class A {  
    A(int i){  
        System.out.print("A");  
    }  
    abstract void f();  
}  
  
class B extends A {  
    void f(){  
        B(){  
            System.out.print("B");  
        }  
    }  
    public static void main(String[] args) {  
        A a = new B();  
    }  
}
```

- A) Errore a tempo di compilazione.
- B) Errore a tempo di esecuzione.
- C) Stampa: A
- D) Stampa: B
- E) Nessuna delle precedenti.

### Versione 12 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public abstract class A {  
    A(int i){  
        System.out.print("A");  
    }  
    abstract void f();  
}  
  
class B extends A {  
    void f(){  
        B(){  
            System.out.print("B");  
        }  
    }  
    public static void main(String[] args) {  
        A a = new B();  
    }  
}
```

- A) Errore a tempo di compilazione.
- B) Errore a tempo di esecuzione.
- C) Stampa: A
- D) Stampa: B
- E) Nessuna delle precedenti.

### Versione 13 dell'esercizio 6

Qual è il risultato della compilazione ed esecuzione del seguente programma?

```
public abstract class X {  
    X(){  
        System.out.print("X");  
    }  
    abstract void z();  
}  
  
class Y extends X {  
    void z(){  
        Y(){  
            System.out.print("Y");  
        }  
    }  
    public static void main(String[] args){  
        X x = new Y();  
    }  
}
```

- A) X
- B) Y
- C) Errore a tempo di compilazione.
- D) Errore a tempo di esecuzione.
- E) Nessuna delle precedenti.

### Versione 14 dell'esercizio 6

Qual è l'output di questo codice?

```
interface albero { int f(int i);}
class Pioppo implements albero {
    String tipo="Sempre Verde";
    public int f(int n)
    {
        System.out.println("Pioppo");
        int num=n+1;
        return num;
    }
    public static void main(String args[])
    {
        Pioppo p= new Pioppo();
        System.out.println(p.f(5));
    }
}
```

- A) Errore a tempo di compilazione
- B) Errore a tempo di esecuzione
- C) Pioppo  
6
- D) Pioppo  
1
- E) Nessuna delle precedenti

### **Versione 15 dell'esercizio 6**

Qual è tra le seguenti dichiarazioni non produce un errore di compilazione?

```
public interface I {  
1.     byte b;  
2.     private final int i = 1;  
3.     public static final boolean b;  
4.     static float f = 3;  
5.     protected double m();  
}
```

- A) Quella alla linea 1.
- B) Quella alla linea 2.
- C) Quella alla linea 3.
- D) Quella alla linea 4.
- E) Quella alla linea 5.

### **Versione 16 dell'esercizio 6**

Se nella classe A è dichiarato il metodo

```
protected float f(){return 1.0f;}
```

indicare quale delle seguenti ulteriori dichiarazioni causerà un errore di compilazione in una sottoclassa di A.

- A) `public void f(){}`
- B) `public float f(){return 3.0f;}`
- C) `public void f(double d){}`
- D) `public int f(byte b) throws  
Exception {return b;}`
- E) Nessuna dei precedenti.

### Versione 17 dell'esercizio 6

Qual è l'output di questo codice?

```
class Animale {  
  
    String famiglia;  
    protected Animale (String f){  
        famiglia = f;  
    }  
  
    protected void stampa() {  
        System.out.println("Animale: "  
+ famiglia);  
    }  
}  
  
class Mammifero extends Animale {  
    public final Mammifero(String f){  
        super(f);  
    }  
  
    public void stampa() {  
        System.out.println("Mammifero: "  
+ famiglia);  
    }  
    public static void main(String[] args){  
        Animale a = new Mammifero("Canidi");  
        a.stampa();  
    }  
}
```

- A) Animale: Canidi
- B) Mammifero: Canidi
- C) Errore a tempo di compilazione
- D) Errore a tempo di esecuzione
- E) Nessuna delle precedenti

### Versione 18 dell'esercizio 6

Cosa succede compilando e, se possibile, eseguendo questo codice

```
class C {  
    static int x;  
    C() { x=1; }  
    public static void m() {  
        System.out.println("x di C vale "+x);  
    }  
}  
  
class D extends C {  
    int x = 2;  
    public void m() {  
        System.out.println("x di D vale "+x);  
    }  
}  
  
public class Test {  
    int x = 3;  
    public static void main(String[] a) {  
        C y = new D();  
        y.m();  
    }  
}
```

- A) Errore a tempo di compilazione
- B) Stampa x di C vale 1
- C) Stampa x di C vale 2
- D) Stampa x di D vale 1
- E) Stampa x di D vale 2

### **Versione 19 dell'esercizio 6**

Dato un file contenente il seguente codice:

```
package pk;

public class C {
    public static class I {
    }
}
```

Dire quale delle seguenti affermazioni è vera:

- A) Un'istanza della classe I può essere costruita solo all'interno della classe C.
- B) Un'istanza della classe I può essere costruita solo all'interno del pacchetto pk.
- C) All'interno del pacchetto pk, un'istanza della classe I può essere costruita con `new I()`.
- D) Un'istanza della classe I può essere costruita con `new C.I()`.
- E) Nessuna delle precedenti.

### **Versione 1 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML List.last che restituisce l'ultimo elemento di una lista.

**Versione 2 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML hd che restituisce la testa di una lista.

### **Versione 3 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn (x,L) => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

#### **Versione 4 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn x => fn L => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

### **Versione 5 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fun f x [] = false
  | f x (y::z) = x=y orelse f x z;
```

segnalando eventuali errori.

**Versione 6 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML tl che restituisce la coda di una lista.

### **Versione 7 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML length che calcola la lunghezza di una lista.

### **Versione 8 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML length che calcola la lunghezza di una lista.

### **Versione 9 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML length che calcola la lunghezza di una lista.

**Versione 10 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML tl che restituisce la coda di una lista.

### **Versione 11 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML tl che restituisce la coda di una lista.

**Versione 12** dell'esercizio sui tipi di ML

Scrivere il tipo della funzione ML:

```
fun f x [] = 0
| f x (y::z) = x=y orelse f x z;
```

segnalando eventuali errori.

### **Versione 13 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML List.last che restituisce l'ultimo elemento di una lista.

### **Versione 14 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fun f (x,[]) = false  
| f (x,y::z) = x=y orelse f (x,z);
```

segnalando eventuali errori.

### **Versione 15 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fun f x [] = false
  | f x (y::z) = x=y orelse f x z;
```

segnalando eventuali errori.

### **Versione 16 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn (x,L) => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

### **Versione 17 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn (x,L) => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

### **Versione 18 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fun f x [] = 0
| f x (y::z) = x=y orelse f x z;
```

segnalando eventuali errori.

### **Versione 19 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn (x,L) => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

**Versione 20 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fun f x [] = 0
| f x (y::z) = x=y orelse f x z;
```

segnalando eventuali errori.

### **Versione 21 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fun f x [] = 0
| f x (y::z) = x=y orelse f x z;
```

segnalando eventuali errori.

**Versione 22** dell'esercizio sui tipi di ML

Scrivere il tipo della funzione ML:

```
fn x => fn L => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

### **Versione 23 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML length che calcola la lunghezza di una lista.

### **Versione 24 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML List.last che restituisce l'ultimo elemento di una lista.

**Versione 25** dell'esercizio sui tipi di ML

Scrivere il tipo della funzione ML:

```
fun f x [] = 0
| f x (y::z) = x=y orelse f x z;
```

segnalando eventuali errori.

**Versione 26 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione ML:

```
fn (x,L) => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

### **Versione 27 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML List.nth che restituisce l'i-esimo elemento di una lista.

### **Versione 28 dell'esercizio sui tipi di ML**

Scrivere il tipo della funzione di ML List.nth che restituisce l'i-esimo elemento di una lista.

**Versione 29** dell'esercizio sui tipi di ML

Scrivere il tipo della funzione ML:

```
fn (x,L) => if L=[] then false else let val (y::R) = L in x=y end;
```

segnalando eventuali errori.

### Passaggio parametri versione n. 1

- Scoping dinamico, MODE = OUT per riferimento
- Scoping statico, MODE = IN OUT per copia

```
program p1
int a; int b; int c; int d;
procedure p2([IN OUT x rif] int a)
int d; int b;
procedure p3([IN OUT x rif] int b)
int d;
procedure p4([IN OUT x rif] int d,[MODE] int c)
int b;
BEGIN
b=c-3;
if c>7 then d=d*3 else d=4;
c=4;
a=c;
write(a,b,c,d);
END

BEGIN
d=2;
if c>5 then b=2 else b=1;
c=b+4;
a=3;
p4(a, c);
write(a,b,c,d);
END

BEGIN
d=c-3;
b=1;
a=d;
c=b;
p3(c);
write(a,b,c,d);
END

BEGIN
a=3;
b=2;
c=2;
d=3;
p2(b);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 2

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int q; int r; int s; int t;
procedure p2([IN x copia] int s)
int r; int q;
BEGIN
r=2;
q=s;
s=q+3;
t=s-1;
write(q,r,s,t);
END

procedure p3([MODE] int t)
int r;
BEGIN
r=s*3;
t=s;
q=r*2;
s=s;
p4(s);
write(q,r,s,t);
END

procedure p4([IN OUT x rif] int s)
int r; int q;
BEGIN
r=t-4;
q=1;
s=q*3;
t=1;
p2(q);
write(q,r,s,t);
END

BEGIN
q=2;
r=2;
s=3;
t=1;
p3(s);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 3

- Scoping dinamico, MODE = IN OUT per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x copia] int x)
int t;
procedure p3([IN x copia] int y,[IN x copia] int t)
int z;
procedure p4([IN x copia] int z,[MODE] int y)
int t;
BEGIN
t=y*1;
z=4;
if y<4 then y=y+1 else y=x*4;
if x=9 then x=4 else x=3;
write(x,y,z,t);
END

BEGIN
z=x;
y=3;
t=y-4;
x=y;
p4(t, x);
write(x,y,z,t);
END

BEGIN
t=1;
x=2;
z=4;
y=3;
p3(x, z);
write(x,y,z,t);
END

BEGIN
x=3;
y=3;
z=3;
t=2;
p2(z);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 4

- Scoping dinamico, MODE = OUT per riferimento
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int q; int r; int s; int t;
procedure p2([MODE] int t)
int r; int q;
BEGIN
if s=7 then r=1 else r=t-1;
q=s;
if s=0 then t=4 else t=2;
s=1;
write(q,r,s,t);
END

procedure p3([IN OUT x copia] int q,[IN OUT x copia] int t)
int s;
BEGIN
s=3;
q=t;
t=q*2;
r=4;
p4(r, q);
write(q,r,s,t);
END

procedure p4([IN x copia] int r,[IN OUT x copia] int s)
int t;
BEGIN
t=q+1;
if t<8 then r=4 else r=r;
s=2;
q=3;
p2(t);
write(q,r,s,t);
END

BEGIN
q=4;
r=3;
s=4;
t=0;
p3(r, t);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 5

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int x; int y; int z; int t;
procedure p2([IN x copia] int t)
int z; int x;
procedure p3([IN OUT x copia] int t)
int z;
BEGIN
if y>2 then z=t*2 else z=4;
t=y;
x=4;
y=1;
p4(z);
write(x,y,z,t);
END

BEGIN
z=1;
x=2;
t=x+1;
y=2;
p3(x);
write(x,y,z,t);
END

procedure p4([MODE] int z)
int x; int t;
BEGIN
x=3;
t=3;
z=3;
y=z;
write(x,y,z,t);
END

BEGIN
x=2;
y=1;
z=2;
t=2;
p2(t);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 6

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN per copia

```
program p1
int q; int r; int s; int t;
procedure p2([IN OUT x rif] int r,[IN OUT x rif] int t)
int s;
procedure p3([IN OUT x rif] int t,[IN OUT x copia] int q)
int r;
BEGIN
r=s;
t=1;
q=s;
s=q+3;
p4(q);
write(q,r,s,t);
END

procedure p4([MODE] int t)
int r; int s;
BEGIN
r=t-3;
s=q+2;
t=r+1;
if t<8 then q=r-3 else q=t+3;
write(q,r,s,t);
END

BEGIN
s=4;
r=q;
t=4;
if r<5 then q=q else q=4;
p3(q, t);
write(q,r,s,t);
END

BEGIN
q=3;
r=3;
s=0;
t=4;
p2(t, r);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 7

- Scoping dinamico, MODE = OUT per riferimento
- Scoping statico, MODE = IN per copia

```
program p1
int x; int y; int z; int t;
procedure p2([MODE] int z,[IN OUT x rif] int t)
int y;
BEGIN
if t>10 then y=2 else y=t*1;
t=t-2;
x=3;
p4(t);
write(x,y,z,t);
END

procedure p3([IN OUT x rif] int y,[IN OUT x copia] int z)
int t;
BEGIN
t=4;
if x<8 then y=3 else y=t;
z=y;
x=4;
write(x,y,z,t);
END

procedure p4([IN OUT x rif] int y)
int t; int z;
BEGIN
t=1;
z=x+2;
if x<6 then y=x else y=2;
x=3;
p3(y, z);
write(x,y,z,t);
END

BEGIN
x=1;
y=0;
z=0;
t=1;
p2(x, z);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 8

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per riferimento

```
program p1
int q; int r; int s; int t;
procedure p2([IN OUT x copia] int q)
int r;
procedure p3([MODE] int t,[IN x copia] int r)
int s;
BEGIN
s=3;
t=2;
r=2;
q=1;
p4(q, r);
write(q,r,s,t);
END

procedure p4([IN OUT x rif] int q,[IN OUT x copia] int t)
int s;
BEGIN
s=3;
q=4;
t=2;
r=q*1;
write(q,r,s,t);
END

BEGIN
r=t;
q=2;
if t<2 then s=2 else s=r*3;
if r>8 then t=3 else t=2;
p3(t, q);
write(q,r,s,t);
END

BEGIN
q=4;
r=0;
s=1;
t=4;
p2(t);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 9

- Scoping dinamico, MODE = OUT per riferimento
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int q; int r; int s; int t;
procedure p2([IN x copia] int t)
int r; int s;
BEGIN
r=2;
s=q*4;
t=4;
q=r;
p3(t, s);
write(q,r,s,t);
END

procedure p3([MODE] int r,[IN OUT x copia] int q)
int t;
BEGIN
if r=3 then t=r else t=r-2;
r=1;
q=s;
s=2;
write(q,r,s,t);
END

procedure p4([IN x copia] int s)
int q; int t;
BEGIN
q=r;
t=1;
if r>6 then s=t else s=4;
r=2;
p2(t);
write(q,r,s,t);
END

BEGIN
q=3;
r=4;
s=4;
t=1;
p4(q);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 10

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN per copia

```
program p1
int x; int y; int z; int t;
procedure p2([MODE] int x)
int t; int y;
procedure p3([IN OUT x rif] int y)
int t; int z;
BEGIN
t=3;
z=y-3;
y=3;
x=z;
write(x,y,z,t);
END

procedure p4([IN OUT x copia] int z)
int x; int t;
BEGIN
x=y;
t=y-3;
if t=3 then z=z else z=t;
if t<1 then y=z else y=x+3;
p3(z);
write(x,y,z,t);
END

BEGIN
t=1;
if z<5 then y=4 else y=4;
x=4;
z=x;
p4(t);
write(x,y,z,t);
END

BEGIN
x=2;
y=4;
z=1;
t=1;
p2(t);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 11

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int q; int r; int s; int t;
procedure p2([IN OUT x rif] int t)
int q; int r;
BEGIN
q=s;
r=2;
t=s;
s=t;
p3(r);
write(q,r,s,t);
END

procedure p3([IN x copia] int r)
int t;
BEGIN
t=q*4;
r=4;
q=2;
if r<0 then s=t else s=q;
write(q,r,s,t);
END

procedure p4([IN OUT x copia] int t,[MODE] int r)
int s;
BEGIN
s=4;
t=s;
r=2;
q=s*1;
p2(r);
write(q,r,s,t);
END

BEGIN
q=3;
r=3;
s=0;
t=3;
p4(t, s);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 12

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN OUT per copia

```
program p1
int q; int r; int s; int t;
procedure p2([IN OUT x rif] int t)
int q; int s;
procedure p3([MODE] int r)
int s; int t;
procedure p4([IN x copia] int s)
int t;
BEGIN
t=q;
if s=20 then s=1 else s=r+3;
q=1;
r=2;
write(q,r,s,t);
END

BEGIN
if r=10 then s=2 else s=1;
t=r;
if q<5 then r=4 else r=s*4;
if r=1 then q=r else q=2;
p4(q);
write(q,r,s,t);
END

BEGIN
q=r;
s=2;
t=q;
r=t;
p3(t);
write(q,r,s,t);
END

BEGIN
q=4;
r=2;
s=1;
t=2;
p2(r);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 13

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int a; int b; int c; int d;
procedure p2([IN OUT x copia] int b,[IN OUT x copia] int c)
int d;
BEGIN
d=b;
b=b+3;
c=3;
a=d+3;
write(a,b,c,d);
END

procedure p3([IN OUT x rif] int c)
int a;
procedure p4([MODE] int c)
int b; int d;
BEGIN
if a>20 then b=3 else b=2;
d=3;
c=b;
a=2;
p2(a, d);
write(a,b,c,d);
END

BEGIN
a=d*3;
c=3;
b=c;
d=4;
p4(d);
write(a,b,c,d);
END

BEGIN
a=0;
b=2;
c=2;
d=0;
p3(a);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 14

- Scoping dinamico, MODE = IN per copia
- Scoping statico, MODE = OUT per copia

```
program p1
int a; int b; int c; int d;
procedure p2([MODE] int b,[IN OUT x copia] int c)
int a;
BEGIN
a=b+1;
b=4;
c=2;
d=3;
p3(b, d);
write(a,b,c,d);
END

procedure p3([IN OUT x copia] int c,[IN OUT x rif] int a)
int b;
procedure p4([IN OUT x rif] int d,[IN x copia] int b)
int c;
BEGIN
c=3;
d=a+2;
b=3;
a=3;
write(a,b,c,d);
END

BEGIN
b=a;
c=1;
a=1;
if a>9 then d=2 else d=d*4;
p4(c, d);
write(a,b,c,d);
END

BEGIN
a=4;
b=4;
c=4;
d=0;
p2(c, b);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 15

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int q; int r; int s; int t;
procedure p2([IN OUT x copia] int t)
int r;
BEGIN
r=3;
t=q;
q=1;
s=r+1;
p4(s, r);
write(q,r,s,t);
END

procedure p3([MODE] int s)
int q;
BEGIN
q=t;
s=r*3;
if r<8 then t=r-4 else t=2;
r=t*3;
write(q,r,s,t);
END

procedure p4([IN x copia] int q,[IN OUT x rif] int r)
int t;
BEGIN
t=r*1;
q=3;
r=r*3;
if r=7 then s=2 else s=s-2;
p3(r);
write(q,r,s,t);
END

BEGIN
q=1;
r=2;
s=4;
t=0;
p2(t);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 1

- Scoping dinamico, MODE = IN OUT per copia
- Scoping statico, MODE = OUT per copia

```
program p1
int a; int b; int c; int d;
procedure p2([MODE] int c)
int b; int a;
procedure p3([IN OUT x rif] int c,[IN OUT x rif] int b)
int a;
procedure p4([IN x copia] int d)
int c; int b;
BEGIN
c=3;
b=4;
d=1;
a=4;
write(a,b,c,d);
END

BEGIN
a=d+3;
c=d;
b=c*1;
d=c-3;
p4(c);
write(a,b,c,d);
END

BEGIN
b=2;
a=c;
if a>2 then c=d-3 else c=1;
d=d;
p3(c, d);
write(a,b,c,d);
END

BEGIN
a=3;
b=3;
c=2;
d=2;
p2(a);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 2

- Scoping dinamico, MODE = IN per copia
- Scoping statico, MODE = IN per riferimento

```
program p1
int q; int r; int s; int t;
procedure p2([IN OUT x rif] int s)
int q; int t;
procedure p3([IN x copia] int s,[MODE] int r)
int q;
BEGIN
if s<3 then q=t else q=1;
if q<0 then s=1 else s=q;
r=t-2;
if r<7 then t=4 else t=q-1;
p4(s, r);
write(q,r,s,t);
END

procedure p4([IN OUT x copia] int t,[IN OUT x rif] int s)
int r;
BEGIN
r=4;
t=t;
s=s;
q=2;
write(q,r,s,t);
END

BEGIN
q=r-1;
t=2;
s=1;
r=q;
p3(s, t);
write(q,r,s,t);
END

BEGIN
q=4;
r=3;
s=1;
t=2;
p2(s);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 3

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int q; int r; int s; int t;
procedure p2([MODE] int q)
int r;
procedure p3([IN x copia] int r)
int s;
BEGIN
s=q;
r=4;
q=s+1;
t=s*2;
p4(t);
write(q,r,s,t);
END

BEGIN
r=3;
q=4;
s=r*3;
t=r;
p3(s);
write(q,r,s,t);
END

procedure p4([IN x copia] int s)
int t;
BEGIN
t=4;
s=3;
r=t;
if q<10 then q=4 else q=1;
write(q,r,s,t);
END

BEGIN
q=2;
r=4;
s=4;
t=0;
p2(q);
write(q,r,s,t);
END
```

#### Passaggio parametri versione n. 4

- Scoping dinamico, MODE = IN OUT per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x rif] int x)
int z; int t;
procedure p3([IN OUT x copia] int t,[IN x copia] int z)
int y;
procedure p4([IN x copia] int x,[MODE] int t)
int z;
BEGIN
z=y;
if x>50 then x=t else x=1;
t=2;
y=2;
write(x,y,z,t);
END

BEGIN
if t=5 then y=3 else y=4;
t=2;
z=3;
x=4;
p4(z, t);
write(x,y,z,t);
END

BEGIN
z=2;
t=4;
if y=3 then x=4 else x=2;
y=z-1;
p3(z, t);
write(x,y,z,t);
END

BEGIN
x=0;
y=2;
z=2;
t=0;
p2(x);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 5

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN OUT per copia

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x rif] int y)
int x; int z;
BEGIN
x=t+3;
z=t;
y=3;
t=3;
write(x,y,z,t);
END

procedure p3([IN x copia] int z,[IN OUT x rif] int x)
int y;
procedure p4([IN OUT x rif] int y,[MODE] int z)
int t;
BEGIN
t=2;
t=2;
y=t+3;
if y<5 then z=y-2 else z=4;
x=2;
p2(y);
write(x,y,z,t);
END

BEGIN
if z=2 then y=t else y=2;
z=t;
x=4;
t=y;
p4(x, z);
write(x,y,z,t);
END

BEGIN
x=0;
y=4;
z=0;
t=4;
p3(x, y);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 6

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN OUT per copia

```
program p1
int a; int b; int c; int d;
procedure p2([MODE] int c)
int a; int b;
procedure p3([IN x copia] int d,[IN OUT x copia] int c)
int b;
BEGIN
b=d+2;
if a=10 then d=1 else d=a+1;
c=2;
a=2;
p4(c, a);
write(a,b,c,d);
END

BEGIN
a=2;
b=2;
if d=1 then c=4 else c=3;
d=d-3;
p3(d, c);
write(a,b,c,d);
END

procedure p4([IN OUT x rif] int c,[IN x copia] int d)
int b;
BEGIN
b=c+3;
c=c;
d=1;
a=3;
write(a,b,c,d);
END

BEGIN
a=3;
b=1;
c=0;
d=3;
p2(c);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 7

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int q; int r; int s; int t;
procedure p2([IN OUT x copia] int t)
int s;
BEGIN
if r>20 then s=3 else s=4;
t=4;
if q>2 then r=q*2 else r=r;
if t=2 then q=t else q=s;
write(q,r,s,t);
END

procedure p3([MODE] int s,[IN x copia] int r)
int q;
procedure p4([IN OUT x rif] int t,[IN OUT x copia] int r)
int q;
BEGIN
q=3;
t=4;
r=2;
s=t+3;
p2(s);
write(q,r,s,t);
END

BEGIN
q=s-1;
s=4;
r=r;
t=r;
p4(s, t);
write(q,r,s,t);
END

BEGIN
q=1;
r=4;
s=3;
t=2;
p3(t, q);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 8

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int q; int r; int s; int t;
procedure p2([IN OUT x copia] int t,[IN OUT x copia] int q)
int s;
BEGIN
s=1;
t=3;
q=2;
if s<10 then r=4 else r=4;
p3(q);
write(q,r,s,t);
END

procedure p3([IN x copia] int r)
int q; int s;
BEGIN
if r=4 then q=t+2 else q=2;
s=r+4;
r=1;
t=q-1;
write(q,r,s,t);
END

procedure p4([MODE] int t,[IN OUT x copia] int q)
int r;
BEGIN
r=q;
t=q;
q=2;
s=3;
p2(t, r);
write(q,r,s,t);
END

BEGIN
q=1;
r=1;
s=4;
t=1;
p4(s, t);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 9

- Scoping dinamico, MODE = IN per copia
- Scoping statico, MODE = OUT per riferimento

```
program p1
int a; int b; int c; int d;
procedure p2([MODE] int d,[IN OUT x rif] int b)
int c;
procedure p3([IN OUT x rif] int a,[IN OUT x copia] int b)
int d;
BEGIN
d=3;
a=3;
if d<4 then b=b else b=a+2;
if c=1 then c=4 else c=4;
p4(b, d);
write(a,b,c,d);
END

procedure p4([IN OUT x copia] int b,[IN OUT x rif] int a)
int c;
BEGIN
c=4;
b=4;
if d<5 then a=3 else a=4;
d=3;
write(a,b,c,d);
END

BEGIN
if d<5 then c=a else c=1;
d=d;
b=3;
a=a-4;
p3(a, d);
write(a,b,c,d);
END

BEGIN
a=3;
b=4;
c=0;
d=3;
p2(c, d);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 10

- Scoping dinamico, MODE = IN per copia
- Scoping statico, MODE = OUT per copia

```
program p1
int a; int b; int c; int d;
procedure p2([IN OUT x rif] int d,[IN OUT x copia] int c)
int a;
procedure p3([IN OUT x copia] int d,[MODE] int c)
int b;
BEGIN
b=c+1;
d=c-3;
c=4;
if b>50 then a=c else a=d*2;
p4(b, a);
write(a,b,c,d);
END

procedure p4([IN x copia] int d,[IN x copia] int b)
int c;
BEGIN
c=3;
d=b-2;
b=1;
a=2;
write(a,b,c,d);
END

BEGIN
a=b;
d=a-1;
c=a*3;
b=a-4;
p3(d, c);
write(a,b,c,d);
END

BEGIN
a=0;
b=1;
c=3;
d=3;
p2(b, c);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 11

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int q; int r; int s; int t;
procedure p2([IN x copia] int r)
int s; int q;
BEGIN
s=r*2;
q=t;
r=s+2;
t=q;
p3(r);
write(q,r,s,t);
END

procedure p3([MODE] int s)
int t; int q;
BEGIN
if s=8 then t=4 else t=s*2;
q=s;
s=r+4;
r=s+4;
write(q,r,s,t);
END

procedure p4([IN OUT x copia] int s)
int r;
BEGIN
if s>5 then r=q else r=t;
s=4;
if r=20 then t=t else t=s-4;
q=t;
p2(t);
write(q,r,s,t);
END

BEGIN
q=3;
r=4;
s=4;
t=3;
p4(t);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 12

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int a; int b; int c; int d;
procedure p2([IN OUT x copia] int c)
int a;
BEGIN
a=b+3;
if a>20 then c=b-3 else c=2;
if d=4 then d=b+3 else d=2;
b=d;
p3(d);
write(a,b,c,d);
END

procedure p3([IN x copia] int a)
int d;
procedure p4([MODE] int d,[IN x copia] int a)
int b;
BEGIN
b=d;
if d<4 then d=d-1 else d=2;
a=2;
c=b-3;
write(a,b,c,d);
END

BEGIN
d=4;
a=2;
b=3;
c=3;
p4(d, a);
write(a,b,c,d);
END

BEGIN
a=0;
b=1;
c=3;
d=0;
p2(b);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 13

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN per riferimento

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x rif] int x)
int t; int z;
procedure p3([MODE] int t)
int z; int y;
procedure p4([IN OUT x rif] int y,[IN OUT x rif] int z)
int x;
BEGIN
x=1;
y=2;
z=4;
t=t*4;
write(x,y,z,t);
END

BEGIN
z=4;
y=2;
t=2;
x=2;
p4(t, z);
write(x,y,z,t);
END

BEGIN
t=2;
z=2;
if t=50 then x=t else x=t;
y=x;
p3(x);
write(x,y,z,t);
END

BEGIN
x=0;
y=4;
z=3;
t=2;
p2(t);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 14

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN per copia

```
program p1
int x; int y; int z; int t;
procedure p2([MODE] int z,[IN OUT x rif] int x)
int t;
BEGIN
t=x;
z=4;
x=y;
y=z;
p3(x, t);
write(x,y,z,t);
END

procedure p3([IN OUT x rif] int z,[IN OUT x copia] int t)
int x;
BEGIN
x=z;
z=1;
t=3;
y=z;
p4(x);
write(x,y,z,t);
END

procedure p4([IN x copia] int x)
int z;
BEGIN
z=x+4;
x=z+2;
y=4;
if y=6 then t=4 else t=4;
write(x,y,z,t);
END

BEGIN
x=2;
y=3;
z=0;
t=2;
p2(x, y);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 15

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN per copia

```
program p1
int a; int b; int c; int d;
procedure p2([IN OUT x copia] int b)
int d;
BEGIN
if b=5 then d=2 else d=1;
b=d;
a=b;
c=1;
p4(d);
write(a,b,c,d);
END

procedure p3([IN OUT x rif] int a,[IN OUT x rif] int b)
int d;
BEGIN
d=2;
a=b;
b=1;
c=b*3;
write(a,b,c,d);
END

procedure p4([MODE] int b)
int c;
BEGIN
c=a;
b=c;
a=1;
d=d;
p3(c, a);
write(a,b,c,d);
END

BEGIN
a=3;
b=3;
c=2;
d=2;
p2(a);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 16

- Scoping dinamico, MODE = IN OUT per copia
- Scoping statico, MODE = OUT per riferimento

```
program p1
int a; int b; int c; int d;
procedure p2([MODE] int a,[IN OUT x rif] int d)
int c;
procedure p3([IN x copia] int a)
int b;
procedure p4([IN x copia] int a)
int c;
BEGIN
c=d;
a=2;
d=a;
b=d+2;
write(a,b,c,d);
END

BEGIN
b=c;
a=d+4;
d=3;
c=1;
p4(c);
write(a,b,c,d);
END

BEGIN
if b<9 then c=a*2 else c=1;
a=4;
d=d;
b=2;
p3(a);
write(a,b,c,d);
END

BEGIN
a=1;
b=3;
c=1;
d=4;
p2(d, a);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 17

- Scoping dinamico, MODE = IN per copia
- Scoping statico, MODE = IN per riferimento

```
program p1
int a; int b; int c; int d;
procedure p2([IN OUT x rif] int a,[MODE] int d)
int c;
procedure p3([IN OUT x copia] int d)
int c; int a;
BEGIN
c=2;
a=2;
d=1;
b=2;
write(a,b,c,d);
END

procedure p4([IN OUT x copia] int d,[IN OUT x copia] int a)
int c;
BEGIN
c=1;
d=d-3;
a=1;
b=1;
p3(b);
write(a,b,c,d);
END

BEGIN
c=a;
a=4;
d=3;
if c>20 then b=d-4 else b=c;
p4(d, b);
write(a,b,c,d);
END

BEGIN
a=1;
b=4;
c=0;
d=3;
p2(d, a);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 18

- Scoping dinamico, MODE = IN OUT per riferimento
- Scoping statico, MODE = OUT per riferimento

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x copia] int t,[IN OUT x copia] int y)
int z;
procedure p3([IN x copia] int x,[IN OUT x copia] int t)
int z;
procedure p4([MODE] int t)
int z; int x;
BEGIN
z=t+2;
x=t;
t=2;
y=y-1;
write(x,y,z,t);
END

BEGIN
z=1;
x=t;
t=x+2;
y=t-3;
p4(t);
write(x,y,z,t);
END

BEGIN
z=4;
t=1;
if t>6 then y=y*3 else y=z-1;
x=y*3;
p3(x, t);
write(x,y,z,t);
END

BEGIN
x=0;
y=2;
z=2;
t=2;
p2(x, t);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 19

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN per copia

```
program p1
int a; int b; int c; int d;
procedure p2([IN OUT x copia] int b,[IN x copia] int d)
int a;
BEGIN
a=c+3;
b=c;
d=1;
c=3;
write(a,b,c,d);
END

procedure p3([IN x copia] int a)
int c;
BEGIN
c=d-3;
a=b+4;
b=2;
d=3;
p4(b, a);
write(a,b,c,d);
END

procedure p4([MODE] int d,[IN OUT x rif] int c)
int a;
BEGIN
a=1;
d=c+4;
c=b;
b=c+2;
p2(d, b);
write(a,b,c,d);
END

BEGIN
a=2;
b=3;
c=1;
d=1;
p3(c);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 20

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int x; int y; int z; int t;
procedure p2([IN x copia] int x,[IN OUT x copia] int y)
int z;
BEGIN
z=3;
x=4;
y=x;
if t<8 then t=1 else t=t;
p3(z);
write(x,y,z,t);
END

procedure p3([IN x copia] int y)
int z;
BEGIN
z=t;
y=3;
t=1;
x=z-3;
write(x,y,z,t);
END

procedure p4([MODE] int y)
int x; int z;
BEGIN
x=1;
z=1;
if x=50 then y=2 else y=2;
t=1;
p2(z, t);
write(x,y,z,t);
END

BEGIN
x=4;
y=1;
z=4;
t=2;
p4(x);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 21

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN per riferimento

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x rif] int t)
int y; int z;
procedure p3([IN OUT x copia] int t,[IN OUT x rif] int x)
int z;
procedure p4([MODE] int x)
int y;
BEGIN
y=3;
x=2;
t=2;
z=4;
write(x,y,z,t);
END

BEGIN
z=4;
t=z;
if x<50 then x=x else x=4;
y=x-4;
p4(t);
write(x,y,z,t);
END

BEGIN
y=1;
z=x+2;
t=z*3;
x=1;
p3(z, y);
write(x,y,z,t);
END

BEGIN
x=2;
y=4;
z=3;
t=0;
p2(t);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 22

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN per copia

```
program p1
int a; int b; int c; int d;
procedure p2([MODE] int b)
int c;
BEGIN
c=d;
b=a-4;
if d<0 then a=c*2 else a=a*1;
d=a;
p3(c);
write(a,b,c,d);
END

procedure p3([IN OUT x copia] int d)
int c; int a;
procedure p4([IN OUT x rif] int a)
int b; int d;
BEGIN
b=3;
d=3;
a=d;
c=1;
write(a,b,c,d);
END

BEGIN
if d=50 then c=3 else c=2;
if b=10 then a=3 else a=b;
d=4;
b=c*3;
p4(c);
write(a,b,c,d);
END

BEGIN
a=4;
b=1;
c=3;
d=2;
p2(a);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 23

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN per copia

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x rif] int x,[IN OUT x rif] int y)
int z;
BEGIN
z=x+3;
x=3;
y=4;
if x>0 then t=2 else t=4;
p4(z, t);
write(x,y,z,t);
END

procedure p3([IN OUT x copia] int y,[IN OUT x copia] int x)
int z;
BEGIN
z=2;
if x<7 then y=1 else y=z;
x=x;
t=z*4;
write(x,y,z,t);
END

procedure p4([IN x copia] int y,[MODE] int x)
int z;
BEGIN
z=1;
if x>9 then y=t else y=y;
x=z-2;
if z>8 then t=z else t=2;
p3(y, t);
write(x,y,z,t);
END

BEGIN
x=1;
y=3;
z=3;
t=3;
p2(t, z);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 24

- Scoping dinamico, MODE = OUT per riferimento
- Scoping statico, MODE = IN per riferimento

```
program p1
int q; int r; int s; int t;
procedure p2([MODE] int q)
int s; int r;
procedure p3([IN x copia] int q)
int s; int t;
procedure p4([IN x copia] int q)
int s;
BEGIN
s=q-1;
q=r;
if r<9 then t=4 else t=1;
r=r-3;
write(q,r,s,t);
END

BEGIN
s=1;
t=r;
q=4;
r=q-1;
p4(t);
write(q,r,s,t);
END

BEGIN
s=1;
r=4;
q=r-1;
t=2;
p3(t);
write(q,r,s,t);
END

BEGIN
q=2;
r=3;
s=3;
t=0;
p2(t);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 25

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN OUT per copia

```
program p1
int q; int r; int s; int t;
procedure p2([MODE] int r)
int t;
procedure p3([IN OUT x rif] int q,[IN x copia] int r)
int t;
BEGIN
t=q+3;
q=q*4;
r=2;
s=2;
p4(r);
write(q,r,s,t);
END

procedure p4([IN x copia] int q)
int s;
BEGIN
s=2;
if s<0 then q=1 else q=q;
t=3;
r=4;
write(q,r,s,t);
END

BEGIN
t=r*2;
r=2;
q=1;
s=q+3;
p3(s, q);
write(q,r,s,t);
END

BEGIN
q=3;
r=3;
s=2;
t=0;
p2(r);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 26

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int q; int r; int s; int t;
procedure p2([MODE] int q)
int s;
BEGIN
s=2;
q=t*4;
t=s;
r=1;
write(q,r,s,t);
END

procedure p3([IN OUT x rif] int s)
int q; int t;
BEGIN
q=r;
t=s+2;
s=4;
r=2;
p4(q, s);
write(q,r,s,t);
END

procedure p4([IN OUT x rif] int t,[IN x copia] int q)
int r;
BEGIN
if t>3 then r=s-3 else r=2;
t=1;
q=q;
s=q*3;
p2(q);
write(q,r,s,t);
END

BEGIN
q=1;
r=0;
s=0;
t=3;
p3(t);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 27

- Scoping dinamico, MODE = OUT per riferimento
- Scoping statico, MODE = IN per copia

```
program p1
int x; int y; int z; int t;
procedure p2([IN x copia] int z,[IN x copia] int t)
int x;
BEGIN
x=4;
z=3;
t=3;
y=t;
p3(z, y);
write(x,y,z,t);
END

procedure p3([IN OUT x rif] int z,[MODE] int y)
int x;
BEGIN
x=y*2;
z=y+1;
y=y+4;
t=y;
write(x,y,z,t);
END

procedure p4([IN OUT x copia] int y)
int x;
BEGIN
x=t*1;
y=1;
t=3;
if x=1 then z=x+1 else z=3;
p2(x, t);
write(x,y,z,t);
END

BEGIN
x=1;
y=2;
z=3;
t=0;
p4(t);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 28

- Scoping dinamico, MODE = OUT per riferimento
- Scoping statico, MODE = IN per riferimento

```
program p1
int a; int b; int c; int d;
procedure p2([IN OUT x copia] int c,[MODE] int d)
int b;
procedure p3([IN OUT x rif] int d,[IN OUT x rif] int b)
int a;
procedure p4([IN x copia] int a,[IN OUT x copia] int c)
int b;
BEGIN
if c=3 then b=2 else b=a;
a=4;
c=b;
d=d;
write(a,b,c,d);
END

BEGIN
a=2;
d=2;
b=b;
if b>20 then c=b+1 else c=b;
p4(b, c);
write(a,b,c,d);
END

BEGIN
b=2;
c=a+3;
d=1;
a=4;
p3(a, d);
write(a,b,c,d);
END

BEGIN
a=3;
b=2;
c=1;
d=3;
p2(a, b);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 29

- Scoping dinamico, MODE = OUT per riferimento
- Scoping statico, MODE = IN per copia

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x copia] int z,[MODE] int x)
int y;
BEGIN
y=4;
if x<6 then z=t*2 else z=y;
x=3;
if x>1 then t=y*1 else t=2;
write(x,y,z,t);
END

procedure p3([IN OUT x rif] int z,[IN OUT x rif] int t)
int y;
procedure p4([IN OUT x copia] int x)
int z; int y;
BEGIN
if t>10 then z=1 else z=1;
y=x-4;
x=t;
if x<3 then t=1 else t=1;
p2(y, x);
write(x,y,z,t);
END

BEGIN
y=z;
z=x+4;
if y=0 then t=y-1 else t=t+3;
x=3;
p4(x);
write(x,y,z,t);
END

BEGIN
x=0;
y=3;
z=4;
t=2;
p3(z, x);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 1

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN per copia

```
program p1
int q; int r; int s; int t;
procedure p2([IN OUT x rif] int r)
int s; int t;
procedure p3([MODE] int s,[IN OUT x copia] int q)
int t;
BEGIN
t=s-1;
if t=8 then s=q else s=2;
q=t;
r=r-4;
write(q,r,s,t);
END

BEGIN
s=q;
t=q+4;
r=q-2;
q=r;
p3(r, t);
write(q,r,s,t);
END

procedure p4([IN OUT x copia] int r)
int t; int s;
BEGIN
t=1;
s=1;
r=2;
q=t-2;
p2(q);
write(q,r,s,t);
END

BEGIN
q=3;
r=3;
s=1;
t=0;
p4(r);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 2

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int a; int b; int c; int d;
procedure p2([MODE] int d)
int b; int c;
BEGIN
b=3;
if a<1 then c=2 else c=4;
d=d;
a=a;
p4(a, c);
write(a,b,c,d);
END

procedure p3([IN x copia] int c,[IN x copia] int b)
int a;
BEGIN
a=3;
if c=3 then c=c else c=d+4;
b=3;
d=c+4;
write(a,b,c,d);
END

procedure p4([IN x copia] int d,[IN OUT x copia] int b)
int a;
BEGIN
a=2;
d=b;
b=1;
c=b;
p3(b, a);
write(a,b,c,d);
END

BEGIN
a=4;
b=2;
c=3;
d=0;
p2(d);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 3

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN per copia

```
program p1
int q; int r; int s; int t;
procedure p2([MODE] int q,[IN x copia] int s)
int r;
procedure p3([IN OUT x rif] int q,[IN OUT x copia] int s)
int t;
BEGIN
if r>0 then t=1 else t=2;
q=4;
s=r;
r=t;
p4(r);
write(q,r,s,t);
END

BEGIN
if q<50 then r=1 else r=t;
q=t+3;
s=2;
t=s-2;
p3(s, t);
write(q,r,s,t);
END

procedure p4([IN OUT x copia] int s)
int t;
BEGIN
t=1;
s=s*1;
if s=20 then q=1 else q=s;
r=q+2;
write(q,r,s,t);
END

BEGIN
q=1;
r=1;
s=2;
t=2;
p2(r, s);
write(q,r,s,t);
END
```

#### Passaggio parametri versione n. 4

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per riferimento

```
program p1
int x; int y; int z; int t;
procedure p2([MODE] int t)
int y; int x;
BEGIN
y=2;
x=2;
t=3;
z=1;
p3(x);
write(x,y,z,t);
END

procedure p3([IN x copia] int x)
int z;
BEGIN
z=1;
x=t*2;
t=z;
y=2;
write(x,y,z,t);
END

procedure p4([IN x copia] int t,[IN OUT x copia] int y)
int z;
BEGIN
z=x;
t=2;
y=z;
x=x*1;
p2(t);
write(x,y,z,t);
END

BEGIN
x=2;
y=1;
z=2;
t=1;
p4(y, x);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 5

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN OUT per copia

```
program p1
int q; int r; int s; int t;
procedure p2([MODE] int s)
int r; int t;
BEGIN
r=s-1;
t=2;
s=q;
q=t+4;
write(q,r,s,t);
END

procedure p3([IN x copia] int s,[IN x copia] int r)
int q;
BEGIN
q=r;
s=4;
r=3;
t=1;
p2(q);
write(q,r,s,t);
END

procedure p4([IN OUT x rif] int s)
int t;
BEGIN
t=q+2;
s=s;
if t>50 then q=4 else q=3;
r=t;
p3(q, t);
write(q,r,s,t);
END

BEGIN
q=0;
r=4;
s=1;
t=1;
p4(t);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 6

- Scoping dinamico, MODE = IN per copia
- Scoping statico, MODE = OUT per copia

```
program p1
int q; int r; int s; int t;
procedure p2([MODE] int q)
int s;
procedure p3([IN OUT x rif] int q,[IN OUT x rif] int t)
int r;
procedure p4([IN OUT x copia] int r)
int s; int q;
BEGIN
s=4;
q=t+2;
r=4;
t=t+2;
write(q,r,s,t);
END

BEGIN
r=s+2;
q=1;
t=t-3;
s=1;
p4(q);
write(q,r,s,t);
END

BEGIN
s=r;
r=q;
t=3;
p3(s, t);
write(q,r,s,t);
END

BEGIN
q=1;
r=4;
s=3;
t=2;
p2(q);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 7

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int q; int r; int s; int t;
procedure p2([MODE] int t)
int s;
BEGIN
s=t*1;
t=4;
q=3;
r=r;
p3(t);
write(q,r,s,t);
END

procedure p3([IN x copia] int r)
int t;
procedure p4([IN x copia] int t)
int q; int s;
BEGIN
q=t;
s=t*4;
t=q;
r=s*4;
write(q,r,s,t);
END

BEGIN
t=s;
r=4;
if t<3 then q=3 else q=t+3;
s=r;
p4(q);
write(q,r,s,t);
END

BEGIN
q=1;
r=1;
s=2;
t=3;
p2(r);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 8

- Scoping dinamico, MODE = OUT per riferimento
- Scoping statico, MODE = IN OUT per copia

```
program p1
int q; int r; int s; int t;
procedure p2([IN x copia] int q)
int s; int r;
BEGIN
s=q*2;
r=t;
q=t+4;
t=3;
p3(t, s);
write(q,r,s,t);
END

procedure p3([MODE] int r,[IN x copia] int q)
int t;
procedure p4([IN x copia] int t)
int s; int r;
BEGIN
if t<7 then s=1 else s=q;
r=t*2;
t=r;
q=1;
write(q,r,s,t);
END

BEGIN
t=r;
if q=2 then r=1 else r=t;
q=q+1;
s=q;
p4(t);
write(q,r,s,t);
END

BEGIN
q=4;
r=3;
s=3;
t=4;
p2(q);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 9

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN per copia

```
program p1
int a; int b; int c; int d;
procedure p2([IN OUT x rif] int c)
int d;
BEGIN
d=b;
if b=10 then c=a+2 else c=1;
a=c*4;
if a=50 then b=b*3 else b=3;
write(a,b,c,d);
END

procedure p3([MODE] int b)
int d; int c;
procedure p4([IN OUT x copia] int a)
int c;
BEGIN
c=d;
a=d+4;
b=3;
if a=20 then d=b else d=2;
p2(c);
write(a,b,c,d);
END

BEGIN
d=b;
if a>1 then c=3 else c=a;
b=1;
a=c;
p4(c);
write(a,b,c,d);
END

BEGIN
a=3;
b=1;
c=3;
d=2;
p3(c);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 10

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN per riferimento

```
program p1
int a; int b; int c; int d;
procedure p2([MODE] int b)
int a;
procedure p3([IN OUT x rif] int b)
int c; int a;
BEGIN
c=d+3;
a=1;
b=a;
d=a;
p4(d);
write(a,b,c,d);
END

BEGIN
a=c+4;
b=1;
c=3;
if a>6 then d=4 else d=d;
p3(a);
write(a,b,c,d);
END

procedure p4([IN OUT x rif] int a)
int b;
BEGIN
b=3;
a=a-4;
d=2;
c=2;
write(a,b,c,d);
END

BEGIN
a=0;
b=1;
c=2;
d=4;
p2(d);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 11

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN OUT per copia

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x rif] int z)
int y; int x;
BEGIN
y=z+1;
x=3;
z=4;
t=t*2;
write(x,y,z,t);
END

procedure p3([MODE] int x)
int y;
procedure p4([IN x copia] int y)
int z; int x;
BEGIN
z=4;
x=3;
if t>20 then y=y else y=t+3;
t=t*2;
p2(x);
write(x,y,z,t);
END

BEGIN
y=t*2;
x=x*3;
z=4;
t=4;
p4(x);
write(x,y,z,t);
END

BEGIN
x=3;
y=4;
z=1;
t=0;
p3(x);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 12

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int a; int b; int c; int d;
procedure p2([IN OUT x rif] int d,[MODE] int b)
int c;
BEGIN
c=a;
d=a+1;
if d>5 then b=a-3 else b=c;
a=4;
p4(b);
write(a,b,c,d);
END

procedure p3([IN x copia] int c)
int b;
BEGIN
if a=4 then b=1 else b=a;
c=1;
d=c;
a=1;
p2(b, d);
write(a,b,c,d);
END

procedure p4([IN OUT x rif] int c)
int b; int d;
BEGIN
b=c;
d=2;
c=d+3;
a=2;
write(a,b,c,d);
END

BEGIN
a=0;
b=3;
c=3;
d=4;
p3(b);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 13

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN OUT per copia

```
program p1
int a; int b; int c; int d;
procedure p2([MODE] int b)
int d;
BEGIN
d=b;
b=3;
a=d-1;
c=4;
p3(b, c);
write(a,b,c,d);
END

procedure p3([IN OUT x rif] int d,[IN OUT x rif] int c)
int a;
procedure p4([IN x copia] int b)
int d;
BEGIN
d=a+3;
b=c;
a=4;
c=c+2;
write(a,b,c,d);
END

BEGIN
if d=8 then a=3 else a=4;
if b<8 then d=a else d=b+1;
c=d*2;
b=1;
p4(c);
write(a,b,c,d);
END

BEGIN
a=1;
b=1;
c=4;
d=3;
p2(b);
write(a,b,c,d);
END
```

### Passaggio parametri versione n. 14

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN OUT per copia

```
program p1
int q; int r; int s; int t;
procedure p2([MODE] int q)
int t; int s;
procedure p3([IN x copia] int t)
int r;
BEGIN
r=1;
t=q;
q=t-1;
s=1;
write(q,r,s,t);
END

BEGIN
t=q;
if q=4 then s=q+4 else s=q;
q=1;
r=s-1;
p3(t);
write(q,r,s,t);
END

procedure p4([IN OUT x rif] int r)
int q;
BEGIN
q=s-1;
r=r*4;
if s=1 then s=q-4 else s=4;
t=2;
p2(r);
write(q,r,s,t);
END

BEGIN
q=0;
r=3;
s=2;
t=1;
p4(t);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 15

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = IN OUT per copia

```
program p1
int x; int y; int z; int t;
procedure p2([MODE] int z)
int x;
procedure p3([IN x copia] int x,[IN OUT x rif] int y)
int t;
procedure p4([IN x copia] int y)
int t; int z;
BEGIN
t=3;
if x<20 then z=2 else z=x;
y=3;
x=2;
write(x,y,z,t);
END

BEGIN
t=1;
x=4;
y=3;
z=2;
p4(y);
write(x,y,z,t);
END

BEGIN
x=2;
if x<9 then z=t*1 else z=y;
y=z*1;
if x=3 then t=2 else t=z;
p3(x, z);
write(x,y,z,t);
END

BEGIN
x=1;
y=0;
z=4;
t=2;
p2(x);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 16

- Scoping dinamico, MODE = IN per riferimento
- Scoping statico, MODE = OUT per copia

```
program p1
int q; int r; int s; int t;
procedure p2([IN OUT x copia] int s)
int r;
procedure p3([IN OUT x rif] int s)
int q; int t;
BEGIN
q=r;
t=2;
s=4;
r=3;
p4(q);
write(q,r,s,t);
END

procedure p4([MODE] int r)
int s; int q;
BEGIN
s=3;
q=t;
r=1;
t=2;
write(q,r,s,t);
END

BEGIN
if s<20 then r=2 else r=s-3;
s=r;
q=t*2;
t=1;
p3(s);
write(q,r,s,t);
END

BEGIN
q=1;
r=4;
s=3;
t=1;
p2(q);
write(q,r,s,t);
END
```

### Passaggio parametri versione n. 17

- Scoping dinamico, MODE = IN OUT per riferimento
- Scoping statico, MODE = IN per riferimento

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x copia] int z)
int t;
procedure p3([MODE] int z)
int t;
procedure p4([IN x copia] int z)
int y;
BEGIN
y=1;
z=z*4;
t=3;
x=t*1;
write(x,y,z,t);
END

BEGIN
t=4;
if t<6 then z=y-2 else z=4;
y=2;
x=z;
p4(z);
write(x,y,z,t);
END

BEGIN
t=y-4;
z=4;
y=x;
x=1;
p3(t);
write(x,y,z,t);
END

BEGIN
x=1;
y=4;
z=2;
t=1;
p2(y);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 18

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN per riferimento

```
program p1
int x; int y; int z; int t;
procedure p2([IN OUT x rif] int t)
int x; int y;
procedure p3([IN OUT x rif] int t,[IN OUT x rif] int z)
int y;
procedure p4([MODE] int t,[IN OUT x copia] int x)
int z;
BEGIN
z=1;
t=x;
x=z;
y=y;
write(x,y,z,t);
END

BEGIN
y=x;
t=y;
z=t*1;
x=t;
p4(z, x);
write(x,y,z,t);
END

BEGIN
x=t+1;
y=z;
t=y*4;
z=x;
p3(y, z);
write(x,y,z,t);
END

BEGIN
x=0;
y=0;
z=3;
t=2;
p2(x);
write(x,y,z,t);
END
```

### Passaggio parametri versione n. 19

- Scoping dinamico, MODE = OUT per copia
- Scoping statico, MODE = IN per copia

```
program p1
int q; int r; int s; int t;
procedure p2([IN OUT x copia] int s,[MODE] int r)
int q;
procedure p3([IN OUT x rif] int t,[IN x copia] int s)
int r;
BEGIN
r=t-4;
t=s*1;
s=4;
q=r+1;
p4(s);
write(q,r,s,t);
END

BEGIN
if r=1 then q=s else q=t;
s=4;
if q=10 then r=t else r=t*2;
t=1;
p3(r, s);
write(q,r,s,t);
END

procedure p4([IN OUT x copia] int q)
int r; int t;
BEGIN
r=2;
t=3;
q=t;
s=t*1;
write(q,r,s,t);
END

BEGIN
q=4;
r=3;
s=0;
t=3;
p2(s, r);
write(q,r,s,t);
END
```

## UML

Un negozio possiede delle sedi (con un responsabile e un indirizzo) ed è in contatto con dei fornitori. Responsabili e fornitori sono persone, con nome e cognome. Dei responsabili interessa l'anzianità di servizio e dei fornitori la partita iva. Inoltre il negozio rivende prodotti caratterizzati da un codice e da una descrizione. I fornitori forniscono i prodotti a un certo prezzo (fornitori diversi possono fornire lo stesso prodotto a prezzi diversi). Ai prodotti si possono aggiungere e togliere fornitori; per l'aggiunta occorre specificare il prezzo a cui il fornitore vende l'articolo. Al negozio si possono aggiungere e togliere prodotti; per l'aggiunta occorre specificare (tra le altre cose) un fornitore e il prezzo relativo.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente:

Un amministratore invoca il metodo per aggiungere al negozio un nuovo prodotto con codice SM e descrizione “Sapone per le mani”. Il prodotto è fornito dal fornitore SoapSpA (già esistente) al prezzo di €2.49. Il negozio crea il prodotto e invoca il metodo che lo associa al fornitore specificato.

## UML

Un ufficio pubblico ha del personale, di cui interessano nome e stipendio. Il personale è suddiviso in impiegati e dirigenti, e l'ufficio è diretto da un dirigente. L'ufficio ha degli sportelli, ciascuno dei quali è servito da un impiegato. Ogni sportello possiede una coda di prenotazioni, ciascuna delle quali riporta il nome del richiedente e l'ora di arrivo. Le prenotazioni sono servite in ordine FIFO. Lo sportello permette di aggiungere una nuova prenotazione in coda e di eliminare quella di testa.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente:

L'utente elimina il primo elemento dalla coda dello sportello 1 che in quel momento contiene due prenotazioni. Successivamente, alle 11:00, l'utente inserisce una nuova prenotazione relativa a Mario Rossi in quella stessa coda. (Tra le due operazioni la coda non viene modificata.) Il metodo di inserzione è *ricorsivo*.

## UML

Si vuole progettare un gioco che è costituito da personaggi e livelli.

I personaggi hanno un nome e una immagine che li rappresenta (le immagini sono entità complesse che possono essere visualizzate). Vi sono due tipi di personaggi: protagonisti e nemici. I personaggi possono essere visualizzati e possono agire; *i protagonisti agiscono in modo diverso dai nemici*.

I livelli sono 10. Ogni livello è raffigurato da una immagine e possiede un insieme di personaggi attivi. All'insieme si può chiedere il primo elemento e il successivo elemento (per enumerarli tutti; non è richiesto di mostrare le strutture dati per ottenere l'enumerazione). I livelli permettono di eseguire un turno, cioè di far agire tutti i propri personaggi attivi.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente, mostrando tutte le interazioni tra gli oggetti coinvolti.

Il gioco chiede a un livello di eseguire un turno. Il livello chiede all'insieme di personaggi attivi un primo elemento, e riceve il protagonista. Il livello fa agire il protagonista. Poi il livello chiede all'insieme di personaggi il prossimo elemento. Riceve null, per cui il turno termina.

## UML

Si vuole progettare una applicazione per la gestione di ristoranti che possa trattare piu' di un ristorante (il che è utile per le catene di ristorazione). Ogni ristorante ha un certo numero di tavoli (almeno uno) e gestisce delle prenotazioni. Ogni tavolo possiede un suo numero identificativo e una capienza (cioè il numero di posti). Le prenotazioni contengono giorno, ora, uno o più tavoli, e il cliente che ha prenotato. I clienti hanno nome e numero di telefono, e i ristoranti possono reperire velocemente i clienti usando il loro nome come chiave. Infine ogni ristorante ha una sua lista di attesa (che rispetta l'ordine di arrivo) i cui elementi contengono un cliente e il numero di posti richiesto; un cliente può comparire al massimo in una sola lista di attesa. Si possono inserire clienti in coda alle liste di attesa (specificando anche il numero di posti richiesti); in aggiunta, specificando il numero di posti che si sono liberati, si può cercare nella lista il primo cliente (nell'ordine di arrivo) che ha richiesto quel numero di posti.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente, mostrando tutte le interazioni tra gli oggetti coinvolti.

L'operatore chiede al ristorante R1 di cercare nella propria lista d'attesa il primo cliente che ha chiesto un tavolo da 2 posti. Il primo elemento nella lista richiede 4 posti, quindi si passa la richiesta al secondo elemento della lista, che contiene una richiesta di due posti. Il cliente corrispondente (C2) viene restituito all'operatore. La scansione della lista avviene in modo *ricorsivo*.

## UML

Si vuole modellare una rete ferroviaria. I nodi della rete possono essere solamente stazioni o scambi. Tutti i nodi hanno un nome. Inoltre le stazioni hanno un indirizzo mentre gli scambi hanno una coppia di coordinate (latitudine, longitudine). Gli archi sono etichettati con un numero che rappresenta la lunghezza del tratto ferroviario corrispondente. I percorsi sono liste concatenate di nodi della rete. Uno stesso nodo della rete ferroviaria può appartenere a più percorsi. Ai percorsi si può appendere un nuovo nodo.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle classi che mostra come le classi di associazione vengono trasformate in classi semplici per poter essere implementate.

**Esercizio 3:** Disegnare un diagramma delle sequenze per lo scenario seguente, mostrando tutte le interazioni tra gli oggetti coinvolti. **Se un oggetto ha bisogno di un attributo privato di un altro oggetto dovete mostrare come se lo procura.**

Un utente aggiunge il nodo S1 della rete ferroviaria in coda ad un percorso che già contiene il nodo S2.

## UML – Max 13 punti

Si vuole progettare una applicazione gestionale per il mobilificio Poltrone&Cassetti. Ogni negozio Poltrone&Cassetti ha un indirizzo e una lista concatenata di ordini. Ogni ordine ha un numero identificativo e un articolo. Gli articoli hanno una stringa identificativa e possono essere di soli due tipi: interni ed esterni. Quelli interni contengono la fabbrica Poltrone&Cassetti di provenienza, descritta con una stringa. Quelli esterni hanno un fornitore, caratterizzato da nome e indirizzo. Ogni negozio Poltrone&Cassetti fa uso di un database di backend progettato da altri, di cui si sa unicamente che supporta il metodo `query(String q):String`. Dai negozi si può eliminare il primo ordine della lista.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.  
**[max 8 punti]**

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente, mostrando tutte le interazioni tra gli oggetti coinvolti. **Se un oggetto ha bisogno di un attributo privato di un altro oggetto dovete mostrare come se lo procura.**

Un utente rimuove da un negozio N1 il primo ordine O1. La nuova testa della lista di ordini diventa il suo successore O2. L'oggetto O1 viene distrutto.

**[max 6 punti]**

## UML

Si vuole progettare una applicazione gestionale per un ufficio vendite, che possiede degli impiegati caratterizzati da nome, stipendio e ammontare delle vendite. Gli impiegati possono essere venditori o supervisori. Non si possono creare impiegati che non appartengano a una di queste due categorie. I supervisori hanno un bonus; il loro stipendio totale è calcolato sommando l'attributo stipendio con il bonus. Per i venditori lo stipendio coincide con l'attributo omonimo. Un rapporto impiegati è una lista ordinata di impiegati. Un rapporto impiegati è in grado di calcolare la somma delle vendite di tutti i dipendenti che contiene.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente, mostrando tutte le interazioni tra gli oggetti coinvolti. **Se un oggetto ha bisogno di un attributo privato di un altro oggetto dovete mostrare come se lo procura.**

Un utente chiede di calcolare il totale delle vendite dei dipendenti elencati in un rapporto impiegati che contiene due elementi: un venditore che ha venduto 1500€ di merce e un supervisore le cui vendite ammontano a 1600€. Il calcolo avviene in modo ricorsivo sulla lista di impiegati.

## UML

Progettare un sistema di prenotazione viaggi con le seguenti caratteristiche. Un viaggio è una lista concatenata di tratte. Ogni tratta possiede un contratto, rappresentato con una stringa, e un biglietto. I biglietti possono essere solo aerei o ferroviari. Ogni biglietto ha un luogo e una data di partenza e un luogo di arrivo. I biglietti aerei hanno anche un'ora di imbarco, mentre i biglietti ferroviari hanno un codice di sconto rappresentato con una stringa. Ai viaggi si possono aggiungere tratte in coda.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente, mostrando tutte le interazioni tra gli oggetti coinvolti. **Se un oggetto ha bisogno di un attributo privato di un altro oggetto dovete mostrare come se lo procura.**

Un utente crea una nuova tratta via aerea (che a sua volta crea il proprio biglietto), con contratto standard CS, partenza da Napoli il 15/4/2015 e arrivo a Venezia. Poi appende in modo *ricorsivo* la nuova tratta a un viaggio che già contiene una tratta T0.

## UML

Si tratta di realizzare un sito per l'acquisto di libri on line. Il sito può accedere ai libri che distribuisce (caratterizzati da titolo e autore) usando il titolo come chiave. Ogni libro può essere venduto da diverse case editrici (di cui interessano nome e indirizzo), a prezzi diversi.

Il sito possiede dei clienti, di cui mantiene un nickname e una password. Inoltre ogni cliente possiede un carrello della spesa.

Ogni carrello contiene una lista di prodotti, nell'ordine in cui sono stati inseriti. Si può chiedere di calcolare il numero di libri contenuti nel carrello.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente, mostrando tutte le interazioni tra gli oggetti coinvolti. **Se un oggetto ha bisogno di un attributo privato di un altro oggetto dovete mostrare come se lo procura.**

Un cliente aggiunge al proprio carrello il libro *Mattatoio 5*, di Vonnegut. Il carrello già conteneva il libro *Ben Hur*. Il cliente poi chiede quanti libri contiene il carrello e il calcolo viene svolto con un algoritmo *iterativo* accedendo agli oggetti rilevanti (mostrare i messaggi necessari).

## UML

Progettare una applicazione "Calendario". Gli eventi del calendario possono essere solo di due tipi: viaggio e meeting. Tutti gli eventi hanno giorno e ora di inizio e una descrizione. Tutti e tre questi valori sono rappresentati con stringhe. In più i viaggi hanno una durata in giorni, mentre i meeting hanno una durata in minuti. Ogni evento può essere stampato; la stampa è diversa per i due tipi di evento. Il calendario mantiene gli eventi di ciascun (singolo) giorno organizzati in una lista doppiamente concatenata, ordinata rispetto all'ora di inizio. Il calendario può accedere al primo elemento della lista usando il giorno come chiave. Il calendario può stampare gli eventi di un giorno specificato; inoltre può sincronizzare il proprio contenuto con il cloud aziendale mediante il metodo `sync()` che utilizza al suo interno un componente esterno di cui si sa solo che supporta un metodo `syncCloud()`.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente, mostrando tutte le interazioni tra gli oggetti coinvolti. **Se un oggetto ha bisogno di un attributo privato di un altro oggetto dovete mostrare come se lo procura.**

Il calendario stampa la lista concatenata di eventi che contiene gli eventi E1, E2 ed E3. La scansione della lista avviene in modo *iterativo*.

## UML

Si vuole progettare un sistema di avvisi per i monitor di un aeroporto. Comprende dei voli (caratterizzati da un numero di volo, una destinazione, e un orario di partenza rappresentato con una stringa), degli sportelli di check-in (caratterizzati da un numero identificativo e dal numero di terminale in cui si trovano) e dei *gate* (caratterizzati da un numero identificativo e dal terminale in cui si trovano). Inoltre, ogni volo è associato al massimo ad un *gate* (potrebbe non essere ancora noto) e ad un insieme di sportelli in cui si può fare il check-in. Le schermate informative sono di due soli tipi: *check-in* e *imbarco*. Ogni schermata informativa ha un suo messaggio di intestazione e una lista ordinata di voli, da cui si può eliminare il primo elemento e aggiungerne uno in coda. L'operazione di *refresh*, che aggiorna le schermate, mostra informazioni diverse nelle due tipologie di schermata. Il refresh utilizza un componente esterno – di cui si sa solo che supporta un metodo *get\_ads()* – per ricevere le pubblicità da mostrare vicino alle informazioni sui voli.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente, mostrando tutte le interazioni tra gli oggetti coinvolti. **Se un oggetto ha bisogno di un attributo privato di un altro oggetto dovete mostrare come se lo procura.**

Un operatore chiede a una schermata di aggiungere il volo AX123 alla lista, che al momento contiene un volo. Poi l'operatore fa il refresh della schermata (operazione che causa lo scaricamento delle pubblicità). L'aggiunta è implementata in modo *iterativo*.

## UML

Progettare un sistema di supporto al lavoro di ufficio come descritto in seguito. Vi sono dei compiti, ciascuno dei quali ha una descrizione testuale, un responsabile (caratterizzato da nome e email) e una data entro cui il compito deve essere svolto. La data è rappresentata con un intero. Un compito può essere elementare o composito. Quelli compositi rappresentano una sequenza di compiti più semplici; per questo posseggono anche una lista ordinata di compiti (che possono essere elementari o compositi a loro volta). I compiti supportano una operazione di notifica: se la data odierna è maggiore della data di scadenza del compito, allora viene mandata una mail al responsabile per avvisarlo. Il componente che fornisce la data odierna e quello che invia la mail non sono sotto la vostra responsabilità (li sviluppa qualcun altro). Sapete solo che il primo supporta un metodo *get\_date()* che restituisce la data, e che il secondo supporta un metodo *send\_mail(...)* i cui input sono il destinatario e il testo del messaggio.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente, mostrando tutte le interazioni tra gli oggetti coinvolti. **Se un oggetto ha bisogno di un attributo privato di un altro oggetto dovete mostrare come se lo procura.**

L'utente invoca il metodo di notifica sul compito elementare C0, che ha scadenza D0. La data odierna è  $D > D_0$ . Il metodo di notifica legge la data odierna mediante l'apposito componente; poichè è maggiore di  $D_0$ , il metodo di notifica invia la mail al responsabile di C0 mediante l'apposito componente.

## UML

Progettare una applicazione per la gestione di utenze telefoniche con le seguenti caratteristiche. Ogni utente ha un nome e un credito. Inoltre è associato a due liste concatenate di telefonate, quelle in ingresso e quelle in uscita. Tutte le telefonate hanno una durata. In aggiunta, quelle in ingresso hanno il numero chiamante, mentre quelle in uscita hanno il numero chiamato e un costo. Non si possono creare telefonate generiche (ma solo in ingresso o in uscita). Ogni utente è associato a un contratto che specifica le date di inizio e fine del contratto stesso, e il costo per minuto delle chiamate.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente, mostrando tutte le interazioni tra gli oggetti coinvolti.

Un utente aggiunge in coda alla lista di telefonate in uscita una nuova chiamata al numero 333-456789 della durata di 50 secondi. Prima dell'aggiunta la lista contiene già due telefonate. L'aggiunta deve essere eseguita in modo *iterativo*.

## UML

Una fabbrica produce un certo numero di prodotti caratterizzati da un codice e dal prezzo. La fabbrica permette l'inserimento di ordinativi, ciascuno dei quali specifica uno dei prodotti, la quantità richiesta, il cliente che la chiede e l'impiegato che segue l'ordine. La fabbrica permette anche la cancellazione dell'ordinativo più vecchio quando questo è stato completato. Gli ordinativi vengono smaltiti in ordine di arrivo (FIFO) e sono conservati in una lista concatenata. I clienti hanno un nome, un codice fiscale e una partita iva. Gli impiegati hanno un nome, un codice fiscale e uno stipendio.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente:

L'utente inserisce un ordinativo O3 in un momento in cui sono in lavorazione altri due ordinativi, O1 e O2. La creazione dell'oggetto O3 è responsabilità del metodo di inserimento, mentre il prodotto, il cliente e l'impiegato esistono già. L'inserzione è *iterativa*. Successivamente l'utente elimina il primo ordinativo (O1). Il metodo di eliminazione rimuove O1 dalla lista e lo distrugge.

## UML

Progettare un sistema per i trasporti locali. L'azienda possiede un insieme di veicoli che possono essere solamente bus, traghetti o tram. Tutti i veicoli hanno una capienza (numero massimo di passeggeri) e un metodo di stampa che fa cose diverse per i diversi tipi di veicoli. Inoltre l'azienda ha dei dipendenti, con nome e stipendio. L'azienda mantiene una lista ordinata di turni, ciascuno dei quali è specificato da: un orario di inizio, uno di fine (rappresentati con stringhe), un dipendente e il veicolo che pilota. Inoltre si possono contare quanti turni sono nella lista

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente, mostrando tutte le interazioni tra gli oggetti coinvolti. **Se un oggetto ha bisogno di un attributo privato di un altro oggetto dovete mostrare come se lo procura.**

L'azienda invia alla lista dei turni, che ne contiene tre, la richiesta di contare i propri membri.  
ATTENZIONE: L'implementazione deve essere iterativa (non ricorsiva).

## UML

Si vuole progettare un meccanismo di conference call (teleconferenza) per il client di un sistema VOIP. Il client possiede un insieme di contatti, caratterizzati da un id e da un nome, che il client può recuperare usando il nome come chiave. Il client può in ogni momento essere associato con al massimo una conference call. Una conference call possiede una lista concatenata di partecipanti, mantenuta nell'ordine in cui questi sono aggiunti alla teleconferenza. Ogni partecipante è un contatto. Il client utilizza un componente esterno per effettuare la connessione vocale, di cui si sa solo che supporta i metodi *call* e *close*, che prendono un contatto in input e iniziano e terminano la chiamata con quel contatto, rispettivamente.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente:

L'utente chiede al client di creare una conference call (inizialmente vuota). Poi chiede al client il contatto di Mario e lo aggiunge alla lista di partecipanti della conference call.

## UML

Progettare un sistema per il controllo del traffico aereo. Un piano di volo è una lista concatenata di punti di passaggio detti *waypoint*. Alcuni casi particolari di waypoint sono coordinate e aeroporti. Le coordinate sono coppie (*latitudine*, *longitudine*), mentre gli aeroporti sono caratterizzati da un nome e da una coordinata.

Deve essere sviluppato un componente detto “Database” che può accedere agli aeroporti usando il nome come chiave e supporta un metodo *search* che data una stringa restituisce l'aeroporto il cui nome è la stringa data.

Ai piani di volo si possono accodare nuovi waypoint. Nel caso degli aeroporti, l'aggiunta può essere effettuata anche fornendo solo il nome (questo metodo sfrutta il database per recuperare l'aeroporto).

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente:

Un pilota (utente) aggiunge l'aeroporto di Linate (fornendo la stringa “Linate”) a un piano di volo che già contiene gli aeroporti di Capodichino e Roma. Il metodo di aggiunta ottiene Linate chiamando *search*, poi lo inserisce in coda al piano di volo. L'inserzione è ricorsiva.

## UML

Si vuole progettare un sistema di gestione per la piscina del CUS. La piscina ha degli utenti, a cui può accedere usando il nome come chiave. Oltre al nome, ad ogni utente è associata una lista concatenata di ingressi (caratterizzati da data e ora). Agli utenti si può aggiungere un ingresso, che verrà posto in coda alla lista dell'utente. Alcuni utenti sono universitari; questi posseggono anche un numero di matricola. Gli utenti implementano l'interfaccia *Serializable* che ha due metodi: *readObject()* e *writeObject()*.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 3:** Disegnare un diagramma delle sequenze per lo scenario seguente, mostrando tutte le interazioni tra gli oggetti coinvolti. **Se un oggetto ha bisogno di un attributo privato di un altro oggetto dovete mostrare come se lo procura.**

Un amministratore aggiunge l'ingresso con data D1 e ora O1 a un utente che nella sua lista ha già due ingressi precedenti. L'inserimento in coda alla lista è *iterativo*.

## UML

Si vuole progettare una applicazione gestionale per una azienda, che possiede degli impiegati caratterizzati da nome e stipendio. Gli impiegati possono essere tecnici, amministrativi, o manager. Non si possono creare impiegati che non appartengano a una di queste tre categorie. I manager hanno un bonus; il loro stipendio totale è calcolato sommando l'attributo stipendio con il bonus. Per gli altri tipi di impiegato lo stipendio coincide con l'attributo omonimo. Un *rapporto impiegati* è una lista ordinata di impiegati. Un rapporto impiegati è in grado di calcolare la somma degli stipendi totali di tutti i dipendenti che contiene.

**Esercizio 1:** Disegnare un diagramma delle classi per queste specifiche.

**Esercizio 2:** Disegnare un diagramma delle sequenze per lo scenario seguente, mostrando tutte le interazioni tra gli oggetti coinvolti.

Un utente chiede di calcolare il totale degli stipendi dei dipendenti elencati in un rapporto impiegati che contiene due elementi: un tecnico con stipendio 1500, e un manager con stipendio 2000 e bonus 500. Il calcolo avviene in modo *iterativo* sulla lista di impiegati.

**Versione 1 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 2 dell'esercizio su programmazione ML**

Scrivere in ML la funzione map.

**Versione 3 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 4 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 5 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 6 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 7 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 8 dell'esercizio su programmazione ML**

Scrivere in ML la funzione map.

**Versione 9 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

**Versione 10 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 11 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 12 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 13 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 14 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

**Versione 15 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 16 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 17 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 18 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

**Versione 19 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 20 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 21 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

**Versione 22** dell'esercizio su programmazione ML

Scrivere in ML la funzione reduce.

**Versione 23 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 24 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 25 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

**Versione 26 dell'esercizio su programmazione ML**

Scrivere in ML la funzione map.

**Versione 27 dell'esercizio su programmazione ML**

Scrivere in ML una funzione che calcola n fattoriale.

**Versione 28 dell'esercizio su programmazione ML**

Scrivere in ML la funzione reduce.

**Versione 29 dell'esercizio su programmazione ML**

Scrivere in ML la funzione filter.

### **Versione 1 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi positivi di una lista.

### **Versione 2 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che decrementa tutti gli elementi di una lista di interi.

### **Versione 3 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola il minimo degli elementi di una lista.

#### **Versione 4 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola il minimo degli elementi di una lista.

### **Versione 5 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola il prodotto degli elementi di una lista.

### **Versione 6 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che trova l'i-esimo elemento di una lista.

### **Versione 7 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi divisibili per 3 di una lista.

### **Versione 8 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi dispari di una lista.

### **Versione 9 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che decrementa tutti gli elementi di una lista di interi.

### **Versione 10 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi negativi di una lista.

### **Versione 11 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola quante volte un elemento x compare in una lista.

### **Versione 12 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola il prodotto degli elementi di una lista.

### **Versione 13 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi negativi di una lista.

### **Versione 14 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi divisibili per 3 di una lista.

### **Versione 15 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che incrementa tutti gli elementi di una lista di interi.

### **Versione 16 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola il prodotto degli elementi di una lista.

### **Versione 17 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi dispari di una lista.

### **Versione 18 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la somma degli elementi di una lista.

### **Versione 19 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola il minimo degli elementi di una lista.

### **Versione 20 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi pari di una lista.

### **Versione 21 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi divisibili per 3 di una lista.

**Versione 22 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola l'inversa di una lista.

### **Versione 23 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la media degli elementi di una lista.

### **Versione 24 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che divide tutti gli elementi di una lista di interi per la lunghezza della lista.

### **Versione 25 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lunghezza di una lista.

**Versione 26 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che trova l'i-esimo elemento di una lista.

### **Versione 27 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi divisibili per 3 di una lista.

**Versione 28 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che trova l'i-esimo elemento di una lista.

### **Versione 29 dell'esercizio su programmazione Prolog**

Scrivere in Prolog un predicato che calcola la lista degli elementi negativi di una lista.