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 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

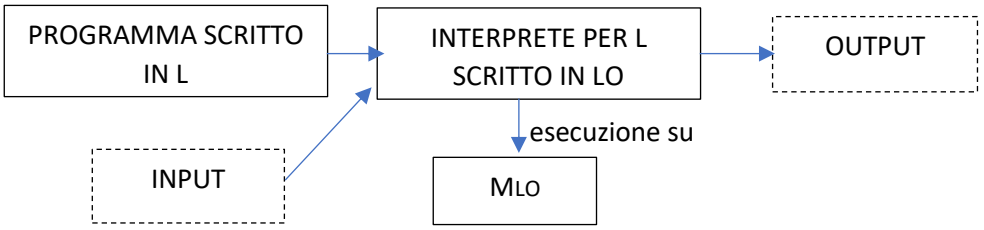
E) Ai metodi static si applica il *dynamic method dispatch*

DOMANDE DI TEORIA

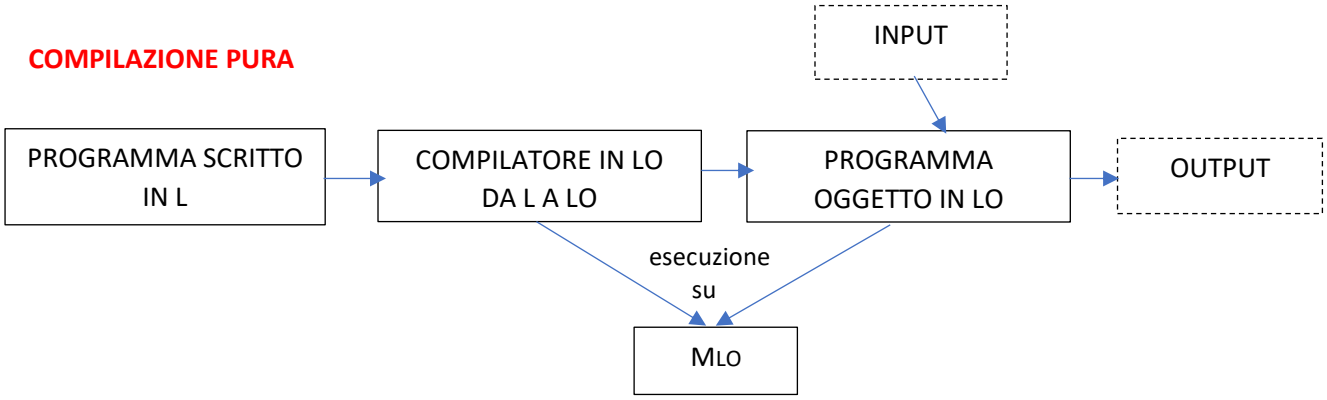
1. Il primo Fortran supportava la ricorsione. **FALSO**
2. Il primo Fortran supportava un'operazione analoga a una malloc. **FALSO**
3. Nel primo Fortran l'occupazione di memoria di un programma era nota a tempo di compilazione. **VERO**
4. Il primo Fortran aveva uno heap e uno stack di attivazione. **FALSO**
5. Il LISP è un linguaggio imperativo. **FALSO**
6. Il Prolog puro supporta i cicli for/next. **FALSO**
7. In Prolog la ricerca del massimo in una lista può essere fatta in forma iterativa (non ricorsiva). **FALSO**
8. Il predicato Prolog member può enumerare i membri di una lista e generare liste. **VERO**
9. SQL è un linguaggio general purpose. **FALSO**
10. HTTP (senza scripts) è un linguaggio general purpose. **FALSO**
11. Un linguaggio puramente funzionale ha i cicli while. **FALSO**
12. Nel paradigma funzionale si possono usare cicli for e while. **FALSO**
13. Nel paradigma logico non si distinguono parametri di input e output. **VERO**
14. Nel paradigma funzionale puro non ci sono gli assegnamenti. **VERO**
15. La funzione membro (elemento, lista) nel paradigma funzionale può essere usata anche per generare tutti gli elementi della lista. **FALSO**
16. Il C++ ha un garbage collector. **FALSO**
17. Il primo garbage collector è stato promosso nel LISP. **VERO**
18. Nei linguaggi funzionali env(x) restituisce il valore di x. **VERO**
19. Nei linguaggi funzionali env(x) restituisce una locazione. **FALSO**
20. In C, l'identificatore x in y=x rappresenta env(x). **FALSO**
21. In C, l'identificatore &x in y=&x rappresenta env(x). **VERO**
22. In un linguaggio funzionale puro, un identificatore x rappresenta env(x). **VERO**
23. Nei linguaggi funzionali l'ambiente associa direttamente gli identificatori al loro valore. **VERO**
24. Nei linguaggi imperativi l'ambiente associa direttamente gli identificatori al loro valore. **FALSO**
25. I linguaggi funzionali non sono computazionalmente completi. **FALSO**
26. In un linguaggio fortemente tipato il controllo dei tipi avviene durante la compilazione e l'esecuzione. **VERO**
27. In un linguaggio staticamente tipato il controllo dei tipi avviene sempre interamente a tempo di compilazione ed esecuzione. **VERO**
28. In un linguaggio dinamicamente tipato il controllo dei tipi avviene interamente a tempo di esecuzione. **VERO**
29. C è debolmente tipato. **VERO**
30. Se il linguaggio è dinamicamente tipato, allora il tipo di una variabile può cambiare durante l'esecuzione del programma. **VERO**
31. Il controllo di correttezza dei downcast richiede controlli a runtime. **VERO**
32. Un linguaggio fortemente e staticamente tipato può avere le union del C. **FALSO**
33. Il polimorfismo per inclusione è il più indicato per la definizione di collezioni di oggetti omogenei. **FALSO**
34. Se in Java si usa unicamente il polimorfismo parametrico allora tutti i controlli di tipo avvengono a tempo di compilazione. **VERO**
35. Il polimorfismo che permette più controlli a tempo di compilazione è quello parametrico. VERO
36. In C, il sistema dei tipi adotta solo la name equivalence. **FALSO**
37. In C, il sistema dei tipi adotta solo la structural equivalence. **FALSO**
38. In C, il sistema dei tipi adotta la name equivalence e la structural equivalence. **VERO**
39. Il C adotta sempre la structural equivalence tra tipi. **FALSO**
40. Il C adotta la structural equivalence per le struct e la name equivalence per tutti gli altri tipi. **FALSO**
41. Si può accedere alle variabili non locali di una procedura in tempo costante, indipendentemente da quanti record di attivazione di devono attraversare. **VERO**
42. La JVM può caricare classi da host diversi. **VERO**
43. Il codice oggetto deve essere eseguito da un interprete diverso dalla macchina hardware. **FALSO**
44. Il polimorfismo parametrico puro può generare errori di tipo a runtime. **FALSO**

45. Le macro hanno un proprio ambiente locale implementato con un record di attivazione. **FALSO**
46. Nei linguaggi a oggetti l'ambiente non locale di un metodo si può trovare: **NELLO HEAP** e **NELLA ZONA STATICA**.
47. La dimensione degli oggetti nello heap può essere esponenziale nell'altezza della gerarchia delle classi (ovvero nel numero di superclassi di una data classe): **IN C++**.
48. Il comando new in Java alloca memoria: **NELLO HEAP**.
49. L'ambiente non locale di una classe interna ad un'altra classe si può trovare: **NELLO HEAP** e **NELLA ZONA STATICA**.
50. Un tipo di dato astratto è: **UN TIPO PERFETTAMENTE INCAPSULATO**.
51. Nei linguaggi funzionali env(x) restituisce: **IL VALORE DI X**.
52. In Java l'ambiente non locale dei metodi si può trovare: **NELLO HEAP** e **NELLA ZONA DOVE SONO MEMORIZZATE LE CLASSI**.
53. Quali forme di polimorfismo supporta Java: **AD HOC**, **PER INCLUSIONE** e **PARAMETRICO**.
54. In un programma che usa le union o altre forme di record varianti il controllo dei tipi può essere fatto interamente a tempo di compilazione: **FALSO**.

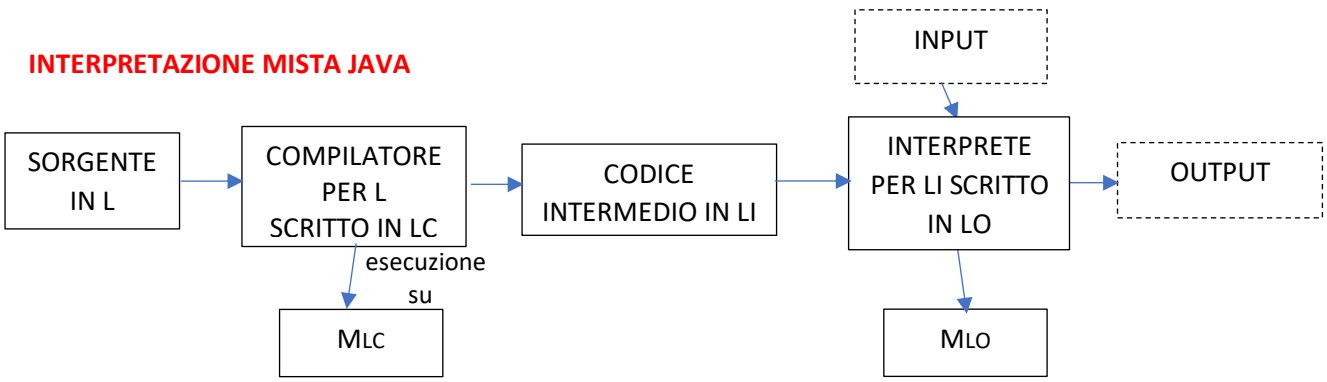
INTERPRETAZIONE PURA



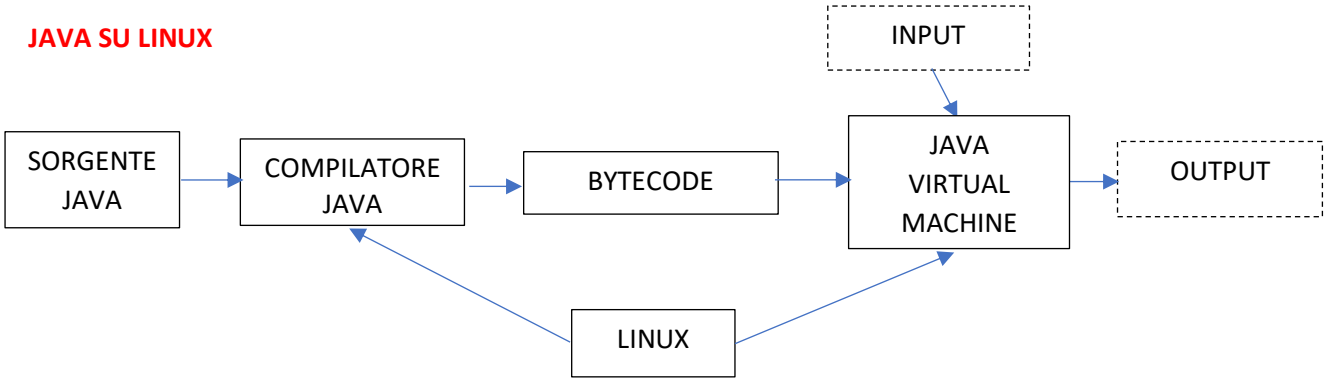
COMPILAZIONE PURA



INTERPRETAZIONE MISTA JAVA



JAVA SU LINUX



1. 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 NO

2. Il primo Fortran supportava un analogo della malloc NO

3. Il Prolog puro supporta i cicli for/next NO

4. Java non ha metodi analoghi alla funzione free() di C e C++ SI

5. Se in Java si usa unicamente il polimorfismo parametrico allora tutti controlli di tipo avvengono a tempo di compilazione. SI

6. Nel paradigma funzionale puro non ci sono gli assegnamenti. SI

7. Nel primo Fortran l'occupazione di memoria di un programma era nota a compile time. SI

8. HTTP (senza scripts) è un linguaggio general purpose. NO

9. Il C++ ha un garbage collector NO

10. La JVM può caricare classi da host diversi SI

11. Un linguaggio puramente funzionale ha i cicli while NO

12. La promozione automatica dei tipi numerici è una forma di polimorfismo ad hoc SI

13. Il C adotta sempre la structural equivalence tra tipi NO

14. Il predicato Prolog member può enumerare i membri di una lista e generare liste SI

15. Tra i compiti del supporto a run time c'è la gestione della memoria SI

16. I linguaggi funzionali non sono computazionalmente completi NO

17. In C++ l'ereditarietà multipla può causare un consumo esponenziale di memoria SI

18. Il primo Fortran supportava la ricorsione. NO

19. Se il linguaggio è dinamicamente tipato, allora il tipo di una variabile può cambiare durante l'esecuzione del programma. SI

20. C adotta la structural equivalence per le struct e la name equivalence per gli altri tipi. NO

21. Si può accedere alle variabili non locali di una procedura in tempo costante, indipendentemente da quanti record di attivazione si devono attraversare. SI

22. I linguaggi funzionali puri non sono computazionalmente completi
NO

23. Compilatore e programmi oggetto sono di norma eseguiti dalla stessa macchina NO

24. C è debolmente tipato SI

25. Nei linguaggi fortemente tipati il type checking avviene tutto a tempo di compilazione NO

SPECIALE

Dire se il polimorfismo parametrico puro (quello dei template):

1. E' più adatto del polimorfismo per inclusione per rappresentare insiemi di oggetti omogenei SI [X] NO [ ]

2. E' più adatto del polimorfismo per inclusione per rappresentare insiemi di oggetti eterogenei SI [ ] NO [X]

3. Ammette il controllo di tipi completo a tempo di compilazione SI [X] NO [ ]

4. Può generare errori di tipo a runtime SI [ ] NO [X]

Esame di LP1

13 Gennaio 2017

Domande generali – Max 8 punti

Esercizio 1: **[2 punti]** Barrare tutte le frasi vere.

1. Il polimorfismo per inclusione è il più indicato per la definizione di collezioni di oggetti omogenei. [ ]
2. Il supporto a run time di un linguaggio puramente funzionale ha sia un ambiente sia una memoria. [ ]

Esercizio 2: **[2 punti]** In C e C++, l'assegnazione le cui parti sinistra e destra denotano *env*(*x*)+1 e *mem*(*env*(*y*))+1, rispettivamente, è:

a) `x[1] = *(y+1);` [ ]
b) `*(x+1) = y+1;` [ ]
c) `x[1] = y+1;` [X]
d) `x[1] = y[1];` [ ]

Esercizio 3: **[2 punti] (Barrare tutte le riposte corrette)** Date le dichiarazioni in C:

```
typedef struct {int a; int b;} Point;

typedef struct {int a; int b;} Pt;

Point v1;

Pt v2;
```

Dire se l'assegnamento:

```
v1 = v2;
```

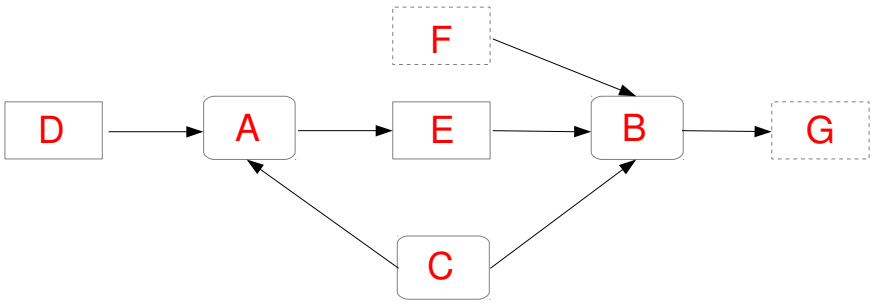
[ ] rispetta la name equivalence [X] rispetta la structural equivalence [ ] è corretto in C

Esercizio 4: **[2 punti]** Il diagramma qui sotto rappresenta l'implementazione di Java su piattaforma Linux. Riempire ognuno dei 7 rettangoli con la lettera corrispondente all'entità che rappresenta.

A – Compilatore Java B – Java virtual machine C – Piattaforma Linux

D – Sorgente Java E – Bytecode F – Input

G – Output



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 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 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) 145666666... (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 ) {
      }
   }
   static void n() {
      try {
         throw( new MyExc2() );
      }
      catch( MyExc1 r ) {
         System.out.print(3);
      }
      finally {
         System.out.print(4);
      }
   }
}
```

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

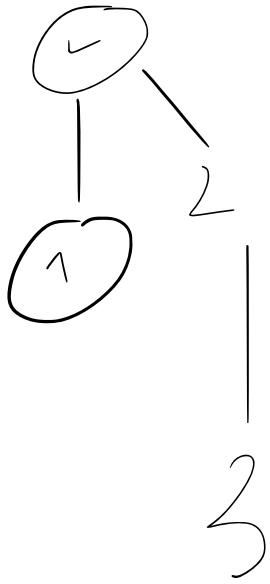
D) 1

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 D1 {
   public static void main(String [] argv)
     throws Exception {
      try {
         q();
         System.out.print(1);
      }
      catch( MyExc2 c ) {
         throw( new MyExc2() );
      }
      finally {
         System.out.print(2);
      }
   }
   static void q()
     throws Exception {
      try {
         throw( new MyExc2() );
      }
   }
}
```



A) Exception in thread "main" MyExc2

B) 2Exception in thread "main" MyExc2

C) Errore a tempo di compilazione

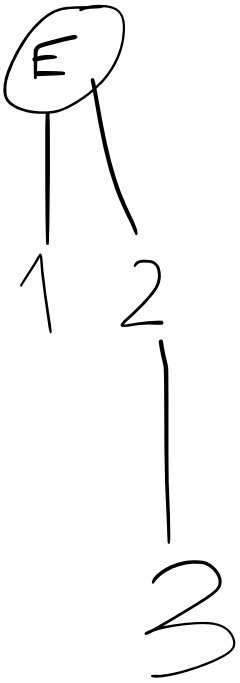
D) ... (ciclo infinito)

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 C1 {
   public static void main(String [] argv)
     throws Exception {
      try {
          System.out.print(1);
          n();
          System.out.print(2);
       }
      catch( Exception s ) {
          System.out.print(3);
       }
       finally {
          System.out.print(4);
          throw( new Exception() );
       }
   }
   static void n()
     throws Exception {
      try {
          throw( new MyExc1() );
       }
     catch( Exception j ) {
          throw( new MyExc3() );
       }
       catch( MyExc2 c ) {
          System.out.print(5);
       }
       catch( MyExc1 d ) {
          System.out.print(6);
          throw( new MyExc2() );
       }
   }
}
```



A) Errore a tempo di compilazione

B) 1324Exception in thread "main" java.lang.Exception

C) 164Exception in thread "main" java.lang.Exception

D) 134Exception 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 MyExc2 { }
public class D1 {
   public static void main(String [] argv)
   throws Exception {
     try {
         m();
      }
      catch( Exception r ) {
         System.out.print(1);
      }
      catch( MyExc1 s ) {
         System.out.print(2);
         throw( new MyExc3() );
      }
      finally {
         System.out.print(3);
      }
   }
   static void m()
   throws Exception {
     try {
         System.out.print(4);
         throw( new Exception() );
      }
      catch( MyExc1 c ) {
         System.out.print(5);
      }
      catch( Exception b ) {
         System.out.print(6);
         throw( new MyExc3() );
      }
      catch( MyExc2 e ) {
      }
      finally {
         System.out.print(7);
         throw( new MyExc3() );
      }
   }
}
```

A) Errore a tempo di compilazione

B) 46713

C) 465753

D) 463

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

A) 145555555... (ciclo infinito)

B) 145Exception in thread "main" MyExc3

C) 1452

D) Errore a tempo di compilazione

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

A) 361

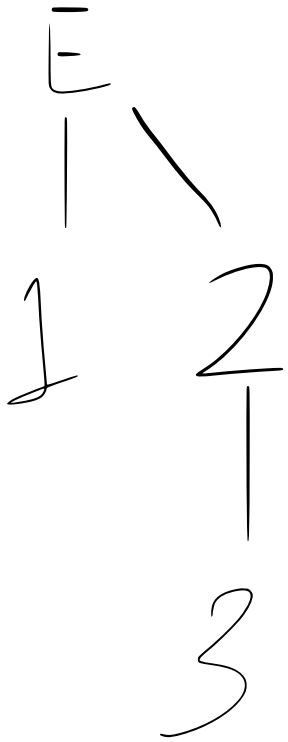
B) 36

C) Errore a tempo di compilazione

D) 36Exception in thread "main" MyExc3

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 C1 {
   public static void main(String [] argv)
     throws Exception {
      try {
         System.out.print(1);
         n();
         System.out.print(2);
      }
      catch( Exception d ) {
         System.out.print(3);
      }
      catch( MyExc3 r ) {
         System.out.print(4);
      }
      finally {
         System.out.print(5);
      }
   }
   static void n()
    throws Exception {
     try {
         System.out.print(6);
         throw( new MyExc1() );
      }
     catch( Exception j ) {
      }
      catch( MyExc2 z ) {
         throw( new MyExc2() );
      }
      catch( MyExc3 v ) {
      }
      finally {
         System.out.print(7);
      }
   }
}
```

A) 1672

B) Errore a tempo di compilazione

C) 16725

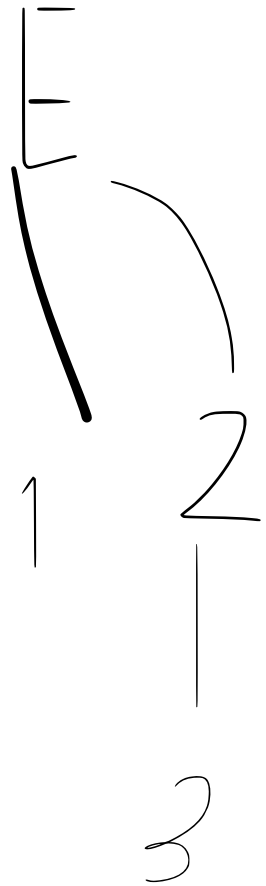
D) 1675Exception in thread "main" MyExc1

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 MyExc2 { }
public class B1 {
   public static void main(String [] argv)
     throws Exception {
      try {
         n();
      }
      catch( MyExc1 z ) {
      }
      catch( Exception g ) {
         System.out.print(1);
      }
      catch( MyExc2 g ) {
      }
   }
   static void n()
     throws Exception {
      try {
         throw( new MyExc3() );
      }
      catch( MyExc2 h ) {
         System.out.print(2);
      }
      catch( MyExc3 h ) {
         System.out.print(3);
      }
      catch( MyExc1 k ) {
      }
      finally {
         System.out.print(4);
      }
   }
}
```



A) 24

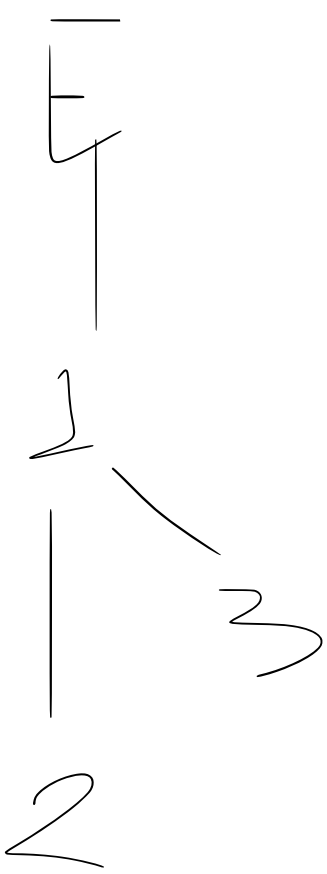
B) 3

C) Errore a tempo di compilazione

D) 34

E) Nessuna delle precedenti

Versione 8 dell'esercizio 5



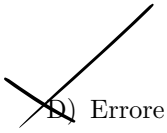
Qual è l'output di questo codice?

```
class MyExc1 extends Exception { }
class MyExc2 extends MyExc1 { }
class MyExc3 extends MyExc1 { }
public class C1 {
   public static void main(String [] argv)
   throws Exception {
      try {
         System.out.print(1);
         q();
         System.out.print(2);
      }
      catch( MyExc2 y ) {
         throw( new Exception() );
      }
      catch( Exception x ) {
      }
      finally {
         System.out.print(3);
      }
   }
   static void q()
     throws Exception {
      try {
         throw( new MyExc1() );
      }
      catch( MyExc1 b ) {
         System.out.print(4);
      }
      catch( Exception s ) {
      }
      catch( MyExc2 t ) {
      }
      finally {
         throw( new MyExc2() );
      }
   }
}
```

A) 143Exception in thread "main" java.lang.Exception

B) 142

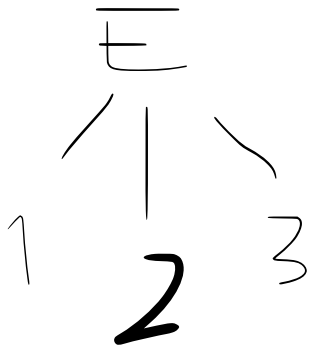
C) 14423



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 Exception { }
public class A1 {
   public static void main(String [] argv)
     throws Exception {
      try {
         System.out.print(1);
         m();
      }
      catch( Exception a ) {
         System.out.print(2);
      }
      catch( MyExc3 a ) {
         throw( new MyExc1() );
      }
      catch( MyExc2 z ) {
      }
   }
   static void m()
     throws Exception {
      try {
         throw( new MyExc1() );
      }
      catch( Exception x ) {
         System.out.print(3);
      }
      finally {
         System.out.print(4);
         throw( new MyExc2() );
      }
   }
}
```

A) 1342

B) 1343

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 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= (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 3 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 4 dell'esercizio 6

Qual è l'output di questo codice?

```
class Stella {
  String nome="Stella";
  final String colore="rosa";

  public Stella(String nome) {
    this.nome = nome;
  }
}

public class Sole extends Stella {

  public Sole(String c) {
    super("Sole");
    this.colore = c;
  }

  public static void main(String [] argv) {
    Sole s = new Sole("bianco");
    System.out.println("Stella: " + s.nome);
    System.out.println("Colore: "
      + s.colore);
  }
```

A) Errore a tempo di compilazione

B) Errore a tempo di esecuzione

C) Stella: Sole
Colore: rosa

D) Stella: Sole
Colore: bianco

E) Nessuna delle precedenti

Versione 5 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 6 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()
   {
System.out.println("Pioppo");
int num=6;
return num;
   }
   public static void main(String args[]) {
       Pioppo p= new Pioppo();
       System.out.println(p.f());
   }
}
```

A) Errore a tempo di compilazione

B) Errore a tempo di esecuzione

C) Pioppo
   6

D) Pioppo
   1

E) Nessuna delle precedenti

Versione 7 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 8 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 9 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 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) 145666666... (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 ) {
      }
   }
   static void n() {
      try {
         throw( new MyExc2() );
      }
      catch( MyExc1 r ) {
         System.out.print(3);
      }
      finally {
         System.out.print(4);
      }
   }
}
```

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

Passaggio parametri versione n. 1

- 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 z)
     int y;
          procedure p3([IN OUT x copia] int x)
          int t;
          BEGIN
          t=y*3;
          if y=50 then x=z*1 else x=t;
          y=y;
          z=2;
          p4(z, t);
          write(x,y,z,t);
          END

          procedure p4([IN OUT x copia] int y,[IN OUT x copia] int t)
          int z;
          BEGIN
          z=x*3;
          y=z+3;
          t=x;
          if y>2 then x=2 else x=1;
          write(x,y,z,t);
          END

     BEGIN
     y=x*1;
     z=4;
     t=x;
     x=1;
     p3(y);
     write(x,y,z,t);
     END

BEGIN
x=4;
y=4;
z=1;
t=2;
p2(x);
write(x,y,z,t);
END
```

Passaggio parametri versione n. 2

- Scoping dinamico, MODE = OUT 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 y)
     int z;
     BEGIN
     z=3;
     y=2;
     t=y+3;
     x=x;
     p3(t);
     write(x,y,z,t);
     END

     procedure p3([IN x copia] int z)
     int x; int t;
          procedure p4([MODE] int x,[IN OUT x copia] int t)
          int z;
          BEGIN
          z=x;
          x=z-3;
          t=4;
          y=2;
          write(x,y,z,t);
          END

     BEGIN
     x=y;
     t=z;
     z=4;
     y=x;
     p4(y, z);
     write(x,y,z,t);
     END

BEGIN
x=3;
y=2;
z=4;
t=3;
p2(y);
write(x,y,z,t);
END
```

Passaggio parametri versione n. 3

- Scoping dinamico, MODE = OUT 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 x)
     int y;
          procedure p3([MODE] int z)
          int y; int t;
          BEGIN
          y=x+4;
          if z>20 then t=z+1 else t=3;
          z=x*4;
          if t<3 then x=z else x=2;
          p4(z);
          write(x,y,z,t);
          END

          procedure p4([IN OUT x copia] int x)
          int y; int t;
          BEGIN
          y=3;
          if x=4 then t=x*1 else t=3;
          x=z*4;
          z=4;
          write(x,y,z,t);
          END

     BEGIN
     y=3;
     x=y;
     t=t;
     z=y;
     p3(y);
     write(x,y,z,t);
     END

BEGIN
x=4;
y=4;
z=2;
t=2;
p2(t);
write(x,y,z,t);
END
```

Passaggio parametri versione n. 4

- Scoping dinamico, MODE = OUT per riferimento
- Scoping statico, MODE = IN per copia

```
program p1
int q; int r; int s; int t;
     procedure p2([IN x copia] int q,[MODE] int s)
     int r;
          procedure p3([IN OUT x copia] int t)
          int s; int r;
          BEGIN
          s=1;
          r=3;
          t=t;
          q=4;
          write(q,r,s,t);
          END

     BEGIN
     if s=7 then r=3 else r=3;
     q=t;
     s=2;
     t=1;
     p3(t);
     write(q,r,s,t);
     END

     procedure p4([IN OUT x rif] int q)
     int t;
     BEGIN
     t=q+2;
     q=3;
     s=r;
     r=r+1;
     p2(s, t);
     write(q,r,s,t);
     END

BEGIN
q=4;
r=1;
s=2;
t=2;
p4(t);
write(q,r,s,t);
END
```

Passaggio parametri versione n. 5

- 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,[IN OUT x rif] int d)
     int b;
          procedure p3([IN x copia] int d)
          int a; int b;
          BEGIN
          a=d-4;
          b=2;
          if c>1 then d=d*4 else d=1;
          c=c*4;
          write(a,b,c,d);
          END

     BEGIN
     b=1;
     c=4;
     d=b+1;
     a=1;
     p3(c);
     write(a,b,c,d);
     END

     procedure p4([IN OUT x rif] int a,[IN OUT x rif] int d)
     int c;
     BEGIN
     if a>2 then c=b else c=1;
     a=d-3;
     d=a+1;
     b=b*1;
     p2(a, d);
     write(a,b,c,d);
     END

BEGIN
a=3;
b=2;
c=0;
d=3;
p4(a, c);
write(a,b,c,d);
END
```

Passaggio parametri versione n. 6

- Scoping dinamico, MODE = OUT per riferimento
- Scoping statico, MODE = IN OUT per riferimento

```
program p1
int a; int b; int c; int d;
     procedure p2([IN x copia] int b)
     int c;
          procedure p3([IN OUT x copia] int a)
          int c;
          BEGIN
          c=d-1;
          a=1;
          d=d-3;
          b=2;
          p4(a);
          write(a,b,c,d);
          END

          procedure p4([MODE] int b)
          int a;
          BEGIN
          a=b;
          b=2;
          d=4;
          c=3;
          write(a,b,c,d);
          END

     BEGIN
     c=d*2;
     b=d;
     d=1;
     a=3;
     p3(c);
     write(a,b,c,d);
     END

BEGIN
a=4;
b=3;
c=4;
d=0;
p2(a);
write(a,b,c,d);
END
```

Passaggio parametri versione n. 7

- 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 d; int b;
          procedure p3([IN OUT x rif] int d,[IN OUT x copia] int b)
          int c;
          BEGIN
          c=d;
          d=1;
          b=a+1;
          a=b;
          write(a,b,c,d);
          END

     BEGIN
     d=3;
     b=2;
     c=a;
     a=b*1;
     p3(c, d);
     write(a,b,c,d);
     END

     procedure p4([IN OUT x rif] int a,[IN OUT x copia] int d)
     int b;
     BEGIN
     b=c*2;
     a=a*3;
     d=d+3;
     c=1;
     p2(d);
     write(a,b,c,d);
     END

BEGIN
a=0;
b=2;
c=4;
d=1;
p4(b, a);
write(a,b,c,d);
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 s;
     BEGIN
     s=1;
     q=2;
     t=q+2;
     if r=9 then r=s+4 else r=r;
     p3(s, q);
     write(q,r,s,t);
     END

     procedure p3([MODE] int t,[IN x copia] int q)
     int s;
          procedure p4([IN OUT x copia] int s)
          int q; int r;
          BEGIN
          q=t+3;
          r=t-2;
          s=s;
          t=q-2;
          write(q,r,s,t);
          END

     BEGIN
     s=3;
     t=s-3;
     q=2;
     r=q*1;
     p4(r);
     write(q,r,s,t);
     END

BEGIN
q=1;
r=2;
s=4;
t=1;
p2(r);
write(q,r,s,t);
END
```

Passaggio parametri versione n. 9

- Scoping dinamico, MODE = IN OUT per riferimento
- Scoping statico, MODE = IN per riferimento

```
program p1
int a; int b; int c; int d;
     procedure p2([IN x copia] int a)
     int d;
          procedure p3([IN x copia] int c)
          int a;
               procedure p4([MODE] int c)
               int a;
               BEGIN
               a=b;
               c=2;
               b=c+1;
               d=c;
               write(a,b,c,d);
               END

          BEGIN
          a=d;
          c=d-2;
          d=4;
          b=b*4;
          p4(a);
          write(a,b,c,d);
          END

     BEGIN
     d=3;
     a=1;
     c=3;
     b=d*2;
     p3(d);
     write(a,b,c,d);
     END

BEGIN
a=4;
b=4;
c=1;
d=1;
p2(c);
write(a,b,c,d);
END
```

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 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 run-time?

   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 run-time?

A) si

B) no

ii) È vero che nel paradigma funzionale puro non ci sono gli assegnamenti?

A) si

B) no