ERRATA CORRIGE
DE
IL NUOVO JAVA

(www.nuovojava.it)

[Versione 4.0 12/2024]

EDITORE ULRICO HOEPLI MILANO

Il nuovo Java

Errata Corrige, ver. 4.0 (28-12-2024)

* Pag. 20 - par. 1.3.2 – terzultima riga della pagina

`Eseguiamo quindi EJE con un doppio clic sul file eje.exe`

Deve essere corretto in:

`Eseguiamo quindi EJE con un doppio clic sul file eje.bat`

* Pag. 37 - par. 1.5.3 – penultima riga della prima nota

`Paragrafo 1.6.2.`

Deve essere corretto in:

`Paragrafo 1.5.2.`

Pag. 58 - par. 2.1.5.1 – ultima riga prima della seconda nota

`unico parametro in input di tipo int, chiamato tasso.`

Deve essere corretto in:

`unico parametro in input di tipo double, chiamato tasso.`

Pag. 70 - par. 2.2.2.1 – seconda riga dopo il primo riquadro di codice

`cambiaTemperatura il quale`

Deve essere corretto in:

`incrementaTemperatura il quale`

Pag. 74 - par. 2.2.2.3 – seconda riga nella prima nota

`(dal numero 3.cc in poi)`

Deve essere corretto in:

`(dal numero 2.cc in poi)`

Pag. 74 - par. 2.2.2.3 – quinta riga dopo la prima nota

`Del metodo sommaDueInteri`

Deve essere corretto in:

`Del metodo somma`

Pag. 107 - par. 3.2.2.2 – prima riga della pagina

`00001001 ([+] 0*26 + 0*25 + 0*24 + 1*23 + 0*22 + 0*21 + 1*20 =`
`+ 0+0+0+8+0+0+1 = +9)`

Deve essere corretto in:

`00001001 ([+] 0*2⁶ + 0*2⁵ + 0*2⁴ + 1*2³ + 0*2² + 0*2¹ + 1*2⁰ =`
`+ 0+0+0+8+0+0+1 = +9)`

Pag. 110 - par. 3.3.1 – ultima istruzione riquadro di codice

```
int n = 0b10100001010001011010000101000101
```

Deve essere corretto in (mancava il ";" finale):

```
int n = 0b10100001010001011010000101000101;
```

Pag. 114 - par. 3.3.2.1 – seconda riga

```
float (tipo a 16 bit)
```

Deve essere corretto in:

```
float (tipo a 32 bit)
```

Pag. 118 - par. 3.3.2 – penultima riga della nota prima nota

```
potete consultare i link 3.3
```

Deve essere corretto in:

```
potete consultare i link 3.4
```

Pag. 118 - par. 3.3.3 – ultima istruzione primo riquadro di codice

```
int n = 0b10100001010001011010000101000101
```

Deve essere corretto in:

```
int n = 0b10100001010001011010000101000101;
```

Pag. 118 - par. 3.3.3 – ultima istruzione secondo riquadro di codice

```
int n = 0b10100001_01000101_10100001_01000101
```

Deve essere corretto in (manca il punto e virgola finale):

```
int n = 0b10100001_01000101_10100001_01000101;
```

Pag. 119 - par. 3.3.4 – prima riga

```
Il tipo di dato boolean utilizza solo un bit per memorizzare un valore e gli
unici valori che può immagazzinare sono true e false.
```

Deve essere corretto in:

```
Il tipo di dato boolean può immagazzinare solo due valori true e false.
```

Pag. 121 - par. 3.3.5.1 – terza riga del secondo riquadro di codice

```
//c ha valore 'B'
```

Deve essere corretto in:

```
//b ha valore 'B'
```

Pag. 128 - par. 3.4.1 – ultime due istruzioni dell'ultimo riquadro di codice

```
data.mese=11 // della variabile dataDiNascita
data.anno=2006
```

Deve essere corretto in (mancano i simboli "*;*"):

```
data.mese = 11; // della variabile dataDiNascita
data.anno = 2006;
```

Pag. 129 - par. 3.4.1 – quinta riga della pagina

```
l'oggetto obj).
```

Deve essere corretto in:

```
l'oggetto dataDiNascita).
```

Pag. 152 - par. 4.1.1 – seconda riga del riquadro di codice

```
int variabile1 = 1;
```

Deve essere corretto in:

```
variabile1 = 1;
```

Pag. 158 - par. 4.1.4 – ultimo riquadro di codice

```
boolean b2 = false == (false)
```

Deve essere corretto in:

```
boolean b2 = false == (false);
```

Pag. 161 - par. 4.1.5 – primo riquadro di codice

```
boolean flag = ( (a != 0) && (b/a > 10) )
```

Deve essere corretto in:

```
boolean flag = ( (a != 0) && (b/a > 10) );
```

Pag. 161 - par. 4.1.5 – secondo riquadro di codice

```
boolean flag = ( (a == 0) || (b/a > 10) )
```

Deve essere corretto in:

```
boolean flag = ( (a == 0) || (b/a > 10) );
```

Pag. 173 - par. 4.3.1.1 – terza riga

```
Per esempio, riprendendo l'array alfabeto che dichiarato
```

Deve essere corretto in:

```
Per esempio, riprendendo l'array alfabeto che abbiamo dichiarato
```

* Pag. 174 - par. 4.3.1.1 – settima riga del primo riquadro di codice

```
} while(i <= 10);
```

Deve essere corretto in:

```
} while(i < 10);
```

Pag. 175 - par. 4.3.2 – terzultima riga del primo riquadro di codice

```
Problema: preparare una tassa di tè
```

Deve essere corretto in:

```
Problema: preparare una tazza di tè
```

* Pag. 178 - par. 4.5.3.2 – ultima riga del riquadro di codice

```
} while(i <= 10);
```

Deve essere corretto in:

```
} while(i < 10);
```

Pag. 185 - par. 4.4.2.2 – settima riga della seconda nota

```
(nel nostro caso VERDE, GIALLO e ROSSO)
```

Deve essere corretto in:

```
(nel nostro caso LUNEDI, MARTEDI, MERCOLEDI, GIOVEDI, VENERDI, SABATO,
DOMENICA)
```

Pag. 196 - par. 4.4.3.3 – settima riga

```
semaforo.cambiaColore(Colore2.NERO);
```

Deve essere corretto in:

```
semaforo.cambiaColore(Colore.NERO);
```

Pag. 198 - par. 4.5.2 – quindicesima riga (prima voce dell'elenco)

```
Problema: preparare una tassa di tè
```

Deve essere corretto in:

```
Problema: preparare una tazza di tè
```

Pag. 226 - par. 5.2.4.3 – settima riga del paragrafo

```
è già osservato che la classe Punto
```

Deve essere corretto in:

```
è già osservato che la classe PuntoFisso
```

Pag. 226 - par. 5.2.4.3 – penultima riga della pagina

```
di cui tratteremo nel prossimo capitolo
```

Deve essere corretto in:

```
che tratteremo nel prossimo capitolo e nel capitolo 8
```

Pag. 232 - par. 5.4 – settima riga del paragrafo

```
profiquo
```

Deve essere corretto in:

```
proficuo
```

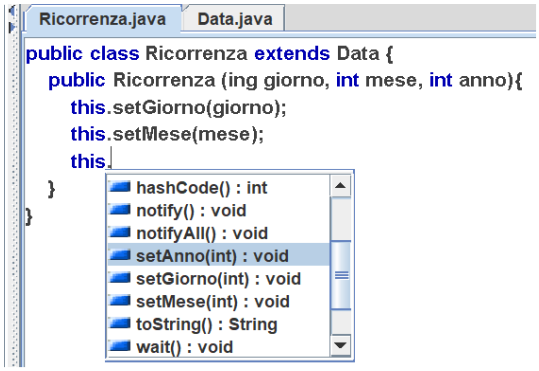
Pag. 250 - par. 6.1.3 – settima riga del primo riquadro di codice

```
public class Studente
```

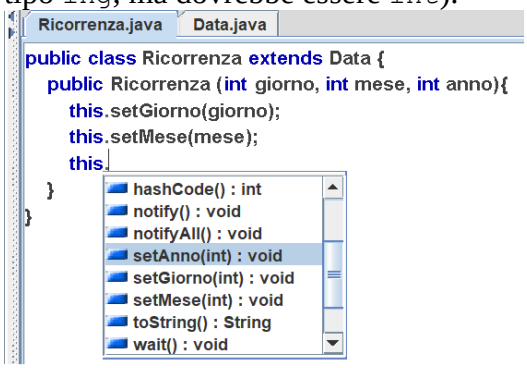
Deve essere corretto in:

```
public class Studente extends Persona
```

Pag. 252 - par. 6.2.1 – figura 6.2



Deve essere cambiata con la seguente immagine (il parametro `giorno` del costruttore è dichiarato di tipo `ing`, ma dovrebbe essere `int`):



* Pag. 254 - par. 6.2.2 – primo riquadro di codice

```
Costruito un Libro!
```

Deve essere corretto in:

```
Costruito un Libro su Java!
```

Pag. 259 - par. 6.2.4.1 – seconda riga

```
di fuori della classe
```

Deve essere corretto in:

```
al di fuori della classe
```

Pag. 297 - par. 7.2.1 – ultima riga del paragrafo

```
1 error
```

Semplicemente non dovrebbe esserci e va eliminato.

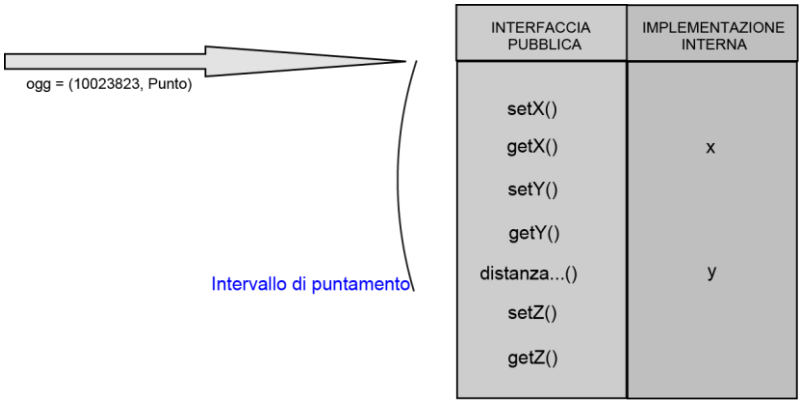
Pag. 306 - par. 7.2.3.3 – primo riquadro di codice della pagina

```
class SuperClass {
    public static void main(String args[]) { }
}

class SubClass extends SuperClass {
    public void main(String args[]) { }
}
```

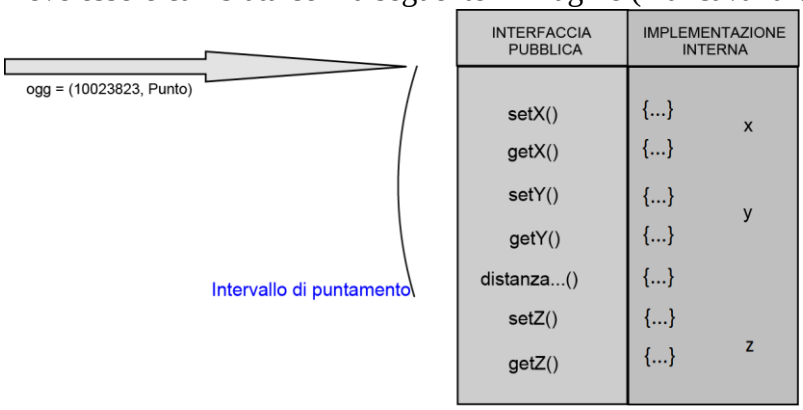
Deve essere corretto in:

```
class SuperClass {
    public void main(String args[]) { }
}

class SubClass extends SuperClass {
    public static void main(String args[]) { }
}
```

Pag. 306 - par. 7.2.3.3 – figura 7.3



Deve essere cambiata con la seguente immagine (mancava la variabile `z`):



Pag. 307 - par. 7.3 – seconda riga della pagina

`costituita dalle dichiarazione`

Deve essere corretto in:

`costituita dalle dichiarazioni`

Pag. 308 - par. 7.3.1 – nona riga del paragrafo

`Per esempio, esiste un override del metodo println`

Deve essere corretto in:

`Per esempio, esiste un overload del metodo println`

Pag. 308 - par. 7.3.2.1 – ultima riga della pagina

`la descrizione delle sua`

Deve essere corretto in:

```
la descrizione delle sue
```

Pag. 320 - par. 7.3.3.3 – prima riga del primo riquadro di codice

```
if (!dip instanceof Programmatore pro) {
```

Deve essere corretto in:

```
if (!(dip instanceof Programmatore pro)) {
```

Pag. 330 - par. 8.3.1.2 – titolo e seconda riga del paragrafo (e nei riferimenti)

```
Helpfull
```

Deve essere corretto in:

```
Helpful
```

Lo stesso refuso si trova anche:

- Nell'indice a pagina XII
- Nella quartultima riga dell'ultimo punto dell'elenco numerato a pagina 206
- Nell'indice analitico a pagina 864

Pag. 382 - par. 9.2.4.1 – titolo

```
Ovveride
```

Deve essere corretto in:

```
Override
```

Pag. 385 - par. 9.2.4.2 – prima riga dopo il primo riquadro di codice

```
Segue l'interfaccia Misura
```

Deve essere corretto in:

```
Segue l'enumerazione Misura
```

Pag. 406 - par. 10.2.1.2 – penultima riquadro di codice della pagina

```
double d2 = i*f-7;
```

Deve essere corretto in:

```
double d2 = i * f - s;
```

Pag. 416 - par. 10.3.1 – ultimo riquadro di codice della pagina

```
ArrayList<Number> list = new ArrayList<Integer>;
```

Deve essere corretto in:

```
ArrayList<Number> list = new ArrayList<Integer>();
```

Pag. 418 - par. 10.3.2 – seconda riga dopo primo riquadro di codice

```
tipi generici
```

Deve essere corretto in:

```
tipi parametro
```

Pag. 421 - par. 10.3.5 – ultima riga del paragrafo

```
Questo fenomeno di deduzione limitata, è nota come wildcard capture.
```

Deve essere corretto in:

```
Questo fenomeno di deduzione limitata, è noto come wildcard capture.
```

Pag. 478 - par. 12.4 – diciassettesima riga del riquadro di codice

```
new Thread(this).start;
```

Deve essere corretto in:

```
new Thread(this).start();
```

Pag. 479 - par. 12.4 – diciannovesima riga del riquadro di codice

```
public void run {
```

Deve essere corretto in:

```
public void run() {
```

Pag. 517 - par. 13.1.7 – terzultima riga del primo riquadro di codice

```
return getNome();
```

Deve essere corretto in:

```
return nome();
```

Pag. 520 - par. 13.1.7 – quinta riga

```
    public List<Film> getFilmFiltrati(FiltroFilm filtroFilm) {
        List<Film> filmFiltrati = new ArrayList<Film>();
        for (Film film : films) {
            if (filtroFilm.filtra(film)) {
                filmFiltrati.add(film);
            }
        }
        return filmFiltrati;
    }
```

Deve essere corretto in:

```
    public List<Film> getFilmDiFantascienza() {
        List<Film> filmDiFantascienza = new ArrayList<Film>();
        for (Film film : films) {
            if ("Fantascienza".equals(film.getGenere())) {
                filmDiFantascienza.add(film);
            }
        }
        return filmDiFantascienza;
    }

    public List<Film> getBeiFilm() {
        List<Film> beiFilms = new ArrayList<Film>();
        for (Film film : films) {
            if (film.getMediaRecensioni() > 3) {
                beiFilms.add(film);
            }
        }
        return beiFilms;
    }
```

Pag. 525 - par. 13.2.2 – prima riga

```
tanto che gli IDE ci consentono di passare
```

Deve essere corretto in:

```
tanto che alcuni IDE ci consentono di passare
```

*Pag. 545 - par. 13.4.4 – seconda riga della nota

```
(BiPredicate, BiSupplier e così via)
```

Deve essere corretto in:

```
(BiPredicate, BiConsumer e così via)
```

*Pag. 559 - par. 14.3 – penultima riga del primo riquadro di codice

```
return opt.orElse("NESSUN TITOLO"));
```

Deve essere corretto in:

```
return opt.orElse("NESSUN TITOLO")).toUpperCase();
```

Pag. 559 - par. 14.3 – penultima riga

```
filter(s -> .getBrand().equals("Samsung")).
```

Deve essere corretto in:

```
filter(s -> s.getMarca().equals("Samsung")).
```

Pag. 562 - par. 14.3.2 – primo riquadro di codice del secondo riquadro di codice

```
public class OggettoModificabile {
    private String nome;
    public OggettoModificabile(String nome) {
        this.nome = nome;
    }
    public void setNome(String nome) {
        this.nome = nome;
    }
    public String getNome() {
        return nome;
    }
    public String toString() {
        return nome;
    }
}
```

Deve essere corretto in:

```
public class OggettoModificabile {
    private String titolo;
    public OggettoModificabile (String titolo){
        this.titolo = titolo;
    }
    public void setTitolo(String titolo) {
        this.titolo = titolo;
    }
    public String getTitolo() {
        return titolo;
    }
    public String toString() {
```

```
        return titolo;
    }
}
```

Pag. 566 - par. 14.4.2 – nota

Le classi IntStream, LongStream, e DoubleStream, definiscono i metodi statici equivalenti a generate che si chiamano rispettivamente ints, longs, e doubles. Tali metodi generano stream infiniti di numeri int, long o double.

Deve essere corretto in:

La classe java.util.Random definisce i metodi statici equivalenti a generate che si chiamano ints, longs, e doubles. Tali metodi generano rispettivamente stream infiniti di numeri int, long o double (ovvero IntStream, LongStream, e DoubleStream).

Pag. 573 - par. 14.6 – seconda riga

qualcosa che sia diversi da uno stream.

Deve essere corretto in:

qualcosa che sia diverso da uno stream.

Pag. 577 - par. 14.6.2.2 – seconda riga primi due riquadri di codice

filter(s -> "Samsung".equals(s.getMarca()).

Deve essere corretto in (manca la parentesi che chiude il metodo `filter`):

filter(s -> "Samsung".equals(s.getMarca())).

Pag. 579 - par. 14.6.2.3 – quarta riga dopo il secondo riquadro di codice

Un'altro interessante utilizzo di Collectors sono i metodi summarizing (summarizingInt, summarizingDouble e summarizingLong) che permettono di ottenere un oggetto di tipo DoubleSummaryStatistics:

Deve essere corretto in (manca la parentesi che chiude il metodo `filter`):

I metodi summarizing di Collectors: summarizingInt, summarizingDouble e summarizingLong, permettono di ottenere rispettivamente oggetti di tipo IntSummaryStatistics, DoubleSummaryStatistics e LongSummaryStatistics

* Pag. 594 - par. 15.3.1 – sesta riga

colorati in grigio.

Deve essere corretto in:

colorati in bianco.

Pag. 594 - par. 15.3.1 – immagine 15.6

Nell'immagine `InputStreamWriter` in realtà è `OutputStreamWriter`.

* Pag. 594 - par. 15.3.1 – sesta riga

colorati in grigio.

Deve essere corretto in:
`colorati in bianco.`

* Pag. 595 - par. 15.3.2 – immagine 15.8

Nell'immagine `PrintOutputStream` in realtà è `PrintStream`.

Pag. 596 - par. 15.3.2 – prima e seconda riga del riquadro di codice

```
InputStream inputStream = new FileInputStream("C:\primoFile");
OutputStream outputStream = new FileOutputStream("C:\secondoFile");
```

Deve essere corretto in (bisogna utilizzare il carattere di escape nella stringa):

```
InputStream inputStream = new FileInputStream("C:\\primoFile");
OutputStream outputStream = new FileOutputStream("C:\\secondoFile");
```

* Pag. 601 - par. 15.4.3 – quartultima riga della pagina

`L'oggetto isr`

Deve essere corretto in :

`L'oggetto ir`

Pag. 608 - par. 15.4.3 – sesta riga

`Per fare ciò, utilizziamo solitamente classi FileOutputStream e ObjectOutputStream`

Deve essere corretto in (manca l'articolo):

`Per fare ciò, utilizziamo solitamente le classi FileOutputStream e ObjectOutputStream`

Pag. 617 - par. 15.4.3.8 – ultimo riquadro di codice

```
  Base64.Encoder encoder = Base64.getMimeEncoder();
      Person p = new Person ("Claudio","De Sio Cesari","xxx");
      try (FileOutputStream fos = new FileOutputStream(
        new File("person.base64"));
        OutputStream os = encoder.wrap(fos);
        ObjectOutputStream oos = new ObjectOutputStream (os);) {
        oos.writeObject (p);
        System.out.println("Object serialized: "+ p);
  }
```

Deve essere corretto in (la classe `Person` in realtà deve essere `Persona` e la formattazione è errata):

```
  Base64.Encoder encoder = Base64.getMimeEncoder();
  Persona p = new Persona ("Claudio","De Sio Cesari","xxx");
  try (FileOutputStream fos = new FileOutputStream(
    new File("person.base64"));
    OutputStream os = encoder.wrap(fos);
    ObjectOutputStream oos = new ObjectOutputStream (os);) {
    oos.writeObject (p);
    System.out.println("Object serialized: "+ p);
  }
```

* Pag. 622 - par. 15.5.2.1 – penultima riga della pagina

```
Il metodo subpath restituisce il numero di elementi che compongono il
path,
```

Deve essere corretto:

```
Il metodo subpath restituisce una sottosezione del path,
```

* Pag. 627 - par. 15.5.2.2 – prima riga della pagina

```
possiamo ottenere anche BufferedInputReader
e BufferedWriter utilizzando i metodi newBufferedInputReader
```

Deve essere corretto in:

```
possiamo ottenere anche BufferedReader
e BufferedWriter utilizzando i metodi newBufferedReader
```

Pag. 632 - par. 15.6.4.1 – diciannovesima riga del riquadro di codice

```
System.err.println("Questo client non riesce a connettersi
```

Deve essere corretto in (mancano le virgolette per delimitare la stringa):

```
System.err.println("Questo client non riesce a connettersi"
```

Pag. 647 - par. 16.2 – prima riga

```
module, open, opens, provides, requires, to, transitive, uses e with
```

Deve essere corretto in:

```
module, exports, open, opens, provides, requires, to, transitive, uses e
with
```

Pag. 651 - par. 16.2.2.1 – quinta riga del paragrafo

```
una funzionalità rilevante del applicazione
```

Deve essere corretto in:

```
una funzionalità rilevante dell’applicazione
```

Pag. 652 - par. 16.2.2.2 – ultima riga del paragrafo

```
bassamente accoppiati con tipi di altre classi
```

Deve essere corretto in:

```
bassamente accoppiati con tipi di altri sottosistemi
```

Pag. 659 - par. 16.2.3.3 – ultima riga della pagina

```
-add-opens nomeModulo/packageDaAprire=nomeModuloClient
```

Deve essere corretto in:

```
--add-opens nomeModulo/packageDaAprire=nomeModuloClient
```

Pag. 660 - par. 16.2.3.4 – titolo, prima e terza riga

```
Provides to
```

Deve essere corretto in:

```
provides with
```

Pag. 664 - par. 16.3.2 – prima riga dopo il riquadro di codice

```
Ma DocumentFactory non importa le implementazioni concrete
```

Deve essere corretto in:

```
Ma Trademarker non importa le implementazioni concrete
```

Pag. 674 – par.16.5.2 – seconda riga

```
è consiglia una implementazione
```

Deve essere corretto in:

```
è consigliata una implementazione
```

Pag. 678 - riepilogo – decima riga

```
provides to
```

Deve essere corretto in:

```
provides with
```

Pag. 691 - par. 17.3 – quart'ultima riga del riquadro di codice

```
} catch (SQLException /*| ClassNotFoundException |*/ IOException e) {
```

Deve essere corretto in:

```
} catch (SQLException /*| ClassNotFoundException*/ | IOException e) {
```

Pag. 693 - par. 17.4.1 – terza e quarta riga dell'ultimo riquadro di codice

```
+ "VALUES (%s,'%s','%s',%s)", album.getAlbumId(), "
+ album.getArtist(), album.getTitle(), album.getReleaseYear());
```

Deve essere corretto in (eliminati + simboli "+" e "," superflui:

```
+ "VALUES (%s,'%s','%s',%s)", album.getAlbumId(),
  album.getArtist(), album.getTitle(), album.getReleaseYear());
```

Pag. 705 – par. 17.5.2 – quattordicesima riga dell'ultimo riquadro di codice

```
savepoint.getSavepointName() +" Savepoint");
```

Deve essere corretto in:

```
bluesSavepoint.getSavepointName() + " Savepoint");
```

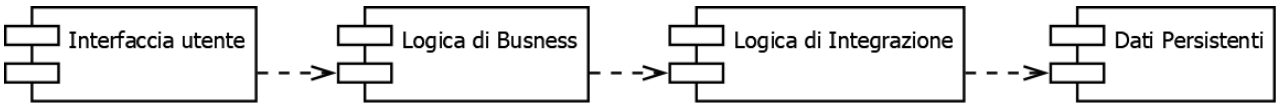
Pag. 710 – par. 17.6.2.1 – terza riga del primo riquadro di codice

```
ResultSet rs = stmt.executeQuery("SELECT a, b FROM TABLE2");
```

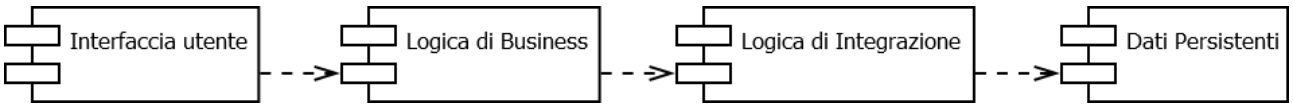
Deve essere corretto in (nome della variabile deve essere coerente con gli snippet successivi):

```
ResultSet res = stmt.executeQuery("SELECT a, b FROM TABLE2");
```

Pag. 714 – par. 17.7 – figura 17.2



Deve essere sostituita con la seguente immagine (manca una "i" in "Business"):



* Pag. 737 – par. 18.1.3 – prima riga metodo main

```
StAXExample stAXExample = new StAXExample();
List<EMail> eMails = stAXExample.leggiFileXML("emails.xml");
```

Deve essere corretto in:

```
StAXExampleReader stAXExampleReader = new StAXExampleReader();
List<EMail> eMails = stAXExampleReader.leggiFileXML("emails.xml");
```

* Pag. 742 – par. 18.1.4.1 – quarta riga prima colonna della tabella

```
expression = "//*[name()!='node1'";
```

Deve essere corretto in manca la parentesi quadra chiusa:

```
expression = "//*[name()!='node1']";
```

Pag. 764 – par. 19.2 – didascalia immagine 19.2

```
Windows XP
```

Deve essere corretto in:

```
Windows 10
```

Pag. 798 – par. 19.5.2.1 – didascalia immagine 19.2

```
Windows XP
```

Deve essere corretto in:

```
Windows 10
```

Pag. 804 – par. 19.5.4.1 – sestultima e quintultima riga della pagina

```
SwingDemo
```

Deve essere corretto in:

```
SwingExample
```

Appendice A - Pag. 7 – par. A.5 – sesta riga del paragrafo

```
Long Term Service (LTS)
```

Deve essere corretto in:

```
Long Term Support (LTS)
```

Appendice F - Pag. 38 – par. F.1 – quarta riga del paragrafo

```
Quesiti metodi
```

Deve essere corretto in:

```
Questi metodi
```

Appendice F - Pag. 50 – par. F.4.1 – settima riga del primo riquadro di codice

```
System.out.println("K"=="K"+local);
```

Deve essere corretto in:

```
System.out.println("KK" == "K" + local);
```

Appendice F - Pag. 60 – par. F.4.3.4 – terza riga della pagina

```
Il metodo String strip() è un'evoluzione del metodo trim introdotto con
Java 11
```

Deve essere corretto in:

```
Il metodo strip introdotto con Java 11, è un'evoluzione del metodo trim
```

Appendice F - Pag. 62 – par. F.4.3.5 – primo riquadro di codice Java

```
String escapedLF = "***\\n***";
```

Deve essere corretto in:

```
String escapedLF = "*** \\n ***";
```

Appendice F - Pag. 67 – par. F.1.2 – ultimo riquadro di codice

```
Integer i = (Integer) list.elementAt(0);
Float p = (Float) list.elementAt(1);
Character c = (Character) list.elementAt(2);
int intero = i.intValue();
float floating = b.floatValue();
character c = c.charValue()
```

Deve essere corretto in:

```
Integer i = (Integer) list.get(0);
Float p = (Float) list.get(1);
Character c = (Character) list.get(2);
int intero = i.intValue();
float floating = p.floatValue();
char character = c.charValue();
```

Appendice F - Pag. 68 – par. F.1.2 – primo riquadro di codice

```
ArrayList<Number> list = new ArrayList<>();
list.add(1);
list.add(false);
list.add('c');
int intero = (Integer) list.elementAt(0);
boolean booleano = (Boolean) list.elementAt(1);
char c = (Character) list.elementAt(2);
```

Deve essere corretto in:

```
ArrayList<Object> list = new ArrayList<>();
list.add(1);
list.add(3.14F);
list.add('c');
int i = (Integer) list.get(0);
float p = (Float) list.get(1);
char c = (Character) list.get(2);
```

* Appendice G - Pag. 90 – par. G.1.4 – sestultima riga della pagina

`Data 31/07/20`
Deve essere corretto in:
`Data 07/31/20`

Appendice H - Pag. 136 – par. H.7.1.2 – primi due punti della lista

- `public static Collection synchronizedCollect`
- `ion(Collection c)`
  Deve essere corretto in:
- `public static Collection synchronizedCollection(Collection c)`

* Appendice I - Pag. 157 – par. I.2.2.1 – sesta riga della pagina

Il termine *compleanno* in realtà deve essere sostituito con *data di nascita*. Di conseguenza nella prima riga del successivo snippet, `mioCompleanno` deve essere sostituito da `dataDiNascita`.

* Appendice I - Pag. 158 – par. I.2.2.4 – prima riga della pagina

Nel titolo del paragrafo `MonthYear` deve essere sostituito con `MonthDay`.

*Appendice M - Pag. 203 – par. M.4 – primo e secondo riquadro con comandi

Bisogna racchiudere la variabile d'ambiente `PATH_TO_FX` tra virgolette:

```
javac --module-path %PATH_TO_FX% --add-modules
```
Deve essere corretto in:
```
javac --module-path "%PATH_TO_FX%" --add-modules
```

NB: il simbolo * precede tutte le voci che sono nuove rispetto alla versione 3.4. Chi aveva già consultato la precedente versione può quindi considerare solo le nuove voci.

L'autore ringrazia tutte le persone che hanno contribuito a realizzare questa errata corrige con le loro segnalazioni via email a claudio@claudiodesio.com:
Emanuele Giuliani, Marco Callegari, Vincenzo Carotenuto, Vincenzo di Michele, Andrea Guida, Enzo Maioli, Daniele Schiappa, Emanuele Caputo, Yuri Palladino, Simone Calugi, Davide Rosignoli, Pierluigi Montinaro, Vincenzo Filangeri, Massimo Bonafede, Stefano Pastori, Cristian Canzini, Roberto Ghitti, Pietro Plazza, Stefano Corsi, Mariarosa Avellino, Gennaro Morelli, Fabio Di Pietro, Giuseppe Sghembri, Simone Berla, Nicholas Cannavacciuolo, Davide Maria Colmi, Sergiu Soroceanu, Vittorio Fasolato, Stefano Taddei, Er, Marcello De Sio Cesari, Luca Giuliani, er mejio, Francesca Piva e Dario Tilesi.

Claudio De Sio Cesari