Errata Corrige di “Manuale di Java 9”

(versione 2)

---

pag. 5 - par. 1.1.1 – diciannovesima riga

memoria ad acceso casuale

Deve essere corretto in:

memoria ad accesso casuale

---

pag. 6 - par. 1.1.3 – prima riga dopo la nota

Esistono tantissimi linguaggi di programmazione e la storia della loro nascita e della loro storia è affascinante

Deve essere corretto in:

Esistono tantissimi linguaggi di programmazione e il racconto della loro storia è affascinante.

---

pag. 20 - par. 1.4.1 – tredicesima riga

Questa metodo

Deve essere corretto in:

Questo metodo

---

pag. 31 - par. 2.1.1 – seconda riga dopo il primo riquadro di codice

I processi (a volte chiamati erroneamente metodologie), di cui parleremo a partire dal quinto paragrafo

Deve essere corretto in:

I processi (a volte chiamati erroneamente metodologie), di cui parleremo a partire dal quinto capitolo

pag. 43 - par. 2.3.1 – quarta riga dopo il riquadro che riporta righe di codice

sui quali approfondiremo il discorso nel capitolo 5.

Deve essere corretto in:

sui quali approfondiremo il discorso nel capitolo 6.

pag. 48 - par. 2.4.1 – sedicesima riga

Il modificatore `final` infatti (che sarà trattato in dettaglio nel capitolo 6)

Deve essere corretto in:

Il modificatore `final` infatti (che sarà trattato in dettaglio nel capitolo 7)

pag. 69 - par. 3.1.1 - secondo riquadro che riporta linee di codice

```
System.out.println(intero + altroIntero);
```

Deve essere corretto in:

```
System.out.println(intero + unAltroIntero);
```

pag. 87 - par. 3.3.2 - terza riga

`NomeClasse.nomeCostanteStatica`

Deve essere corretto in:

`nomeOggetto.nomeCostanteStatica`

pag. 87 - par. 3.3.2 - quinta riga

capitolo 5

Deve essere corretto in:

capitolo 6

pag. 88 - par. 3.3.3 - prima riga e quarta riga

Manca un punto e virgola alla fine dei due statement, quindi alla prima riga:

```
int n = 0b10100001010001011010000101000101
```

Deve essere corretto in:

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

Mentre alla quarta riga:

```
int n = 0b10100001_01000101_10100001_01000101
```

Deve essere corretto in:

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

pag. 91 - par. 3.4 – seconda riga dopo il primo riquadro di codice

e riassegnazione di un altro valore.

Deve essere corretto in:

e dichiarazione di un’altra variabile dello stesso tipo ed assegnazione a questa del tipo precedentemente creato.

pag. 97 - par. 3.5.1 - quarta riga dopo la prima nota

oterremo

Deve essere corretto in:

otterremo

pag. 138 - par. Riepilogo – ventiduesima riga

Per quanto concerne i cicli, di sicuro il più utilizzato è il costrutto `if`

Deve essere corretto in:

Per quanto concerne le condizioni, di sicuro il più utilizzato è il costrutto `if`

pag. 145 - par. 5.1.2.1 – tredicesima riga

In questo caso la variabile booleana `b` è inizializzata automaticamente al suo valore di default `false` automaticamente

Deve essere corretto in:

In questo caso la variabile booleana `b` è automaticamente inizializzata al suo valore di default `false`

pag. 184 - par. 6.2.3 – ventiduesima riga

Figura 6.1 - Class Diagram “Classe Cliente”.

Deve essere corretto in:

Figura 6.1 – Esempio di Class Diagram.

pag. 184 - par. 6.2.3 – terzultima riga

Per quanto riguarda i metodi la sintassi UML:

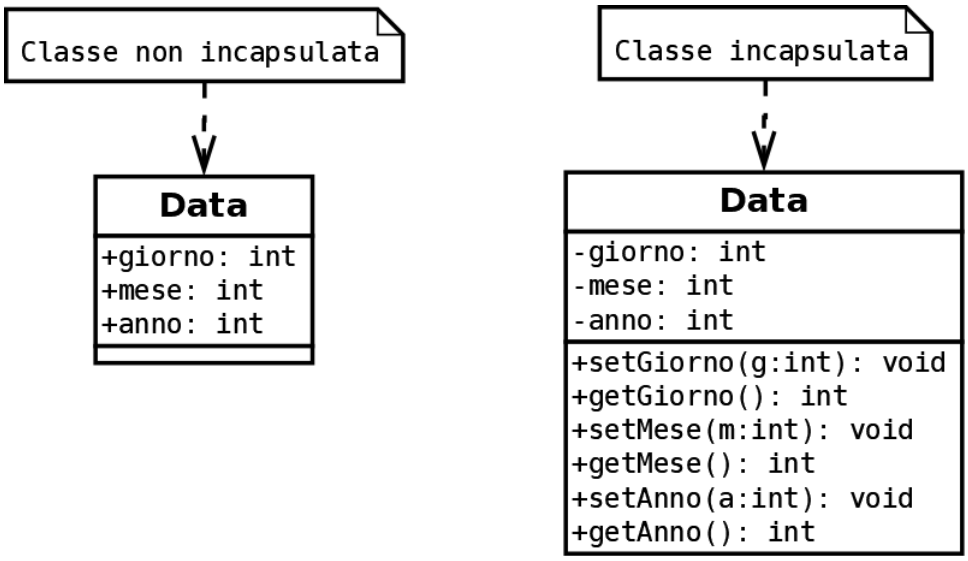
Deve essere corretto in:

Per quanto riguarda i metodi, la sintassi UML è la seguente:

pag. 189 - par. 6.2.3 – Figura 6.2

Nel diagramma delle classi, viene riportato il metodo `getMese(m:int)`

Deve essere corretto eliminando il parametro come nella seguente immagine:



pag. 208 - par. 6.6.2 – prima riga

completamente riorganizzata utilizzando i package.

Deve essere corretto in:

completamente riorganizzata utilizzando i moduli.

pag. 209 - par. 6.7.1 - secondo listato di codice

Vi sono due listati identici al seguente:

```
package com.cdsc.test;

public class PublicInClasse {
    public int variabilePublic;

    public void metodoPublic() {
        System.out.println("Invocato metodo public");
    }
}
```

Il secondo listato deve essere sostituito dal seguente:

```
package com.cdsc;
```

```
import com.cdsc.test.*;

public class PublicClasseFuoriPackage extends PublicInClasse {
    public void metodoCheUsaPublic() {
        metodoPublic();
        System.out.println(variabilePublic);
    }
}
```

pag. 210 - par. 6.7.2 – ottava riga

entrambe le classi dichiarino alcune variabili `stipendio`, `matricola` e

Deve essere corretto in:

entrambe le classi dichiarino le variabili `matricola` e

pag. 213 - par. 6.7.3 – secondo riquadro che riporta righe di codice

La classe deve essere preceduta dalle seguenti dichiarazioni:

```
package com.cdsc;
import com.cdsc.test.*;
```

pag. 219 - par. 6.8.2 – seconda riga

Formattazione errata:

```
non-static variable variabi
leDiIstanza cannot be referenced from a
```

Deve essere corretto in:

```
non-static variable variabileDiIstanza cannot be referenced from a
```

pag. 235 - par. 7.5.1 - subito prima del codice

`Triangolo` e `Rettangolo`. Per esempio:

Deve essere corretto in:

`Triangolo` e `Trapezio`. Per esempio:

pag. 243 - par. 7.5.4 – secondo riga dopo il terzo riquadro che riporta righe di codice

Il nome della superclasse è errato:

`TestPackageEreditarieta`

Deve essere corretto in:

`SuperclasseInPackage`

pag. 243 - par. 7.5.4 – terza riga del quarto riquadro che riporta righe di codice

```
  public class SottoclasseInPackageDiverso extends TestPackageEreditarieta
```

Deve essere corretto in:

```
  public class SottoclasseInPackageDiverso extends SuperclasseInPackage
```

pag. 253 - par. 7.7.1 – prima riga dopo il primo riquadro che riporta righe di codice

In qualsiasi caso ogni membro (variabile o metodo) di una classe astratta ha visibilità pubblica.

Deve essere corretto in:

In qualsiasi caso, ogni membro (variabile o metodo) di un’interfaccia ha visibilità pubblica.

pag. 316 - par. 9.5.1 - terza riga della pagina

questa attributo

Deve essere corretto in:

quest’attributo

pag. 339 - par. 10.1.3 - penultima riga della pagina

```
Auto = new Auto();
```

Deve essere corretto in:

```
Auto auto = new Auto();
```

pag. 343 - par. 10.2.1 – sesta riga del secondo riquadro che riporta righe di codice

```
Integer integer = (String)object;
```

Deve essere corretto in:

```
Integer integer = (Integer)object;
```

pag. 363 - par. 11.1 – titolo del paragrafo

Quando usare le classi innestate

Deve essere corretto in:

Quando usare le classi anonime

pag. 364 - par. 11.1 - ultimo riquadro che riporta linee di codice

```
ContenitoreGenerics<Integer> contenitore2 =
  new ContenitoreGenerics<Integer>();
Contenitore2.setObject(new Integer("1"));
String object = contenitore2.getObject(); //niente cast
System.out.println(object);
```

Deve essere corretto in:

```
ContenitoreGenerics<Integer> contenitore2 =
  new ContenitoreGenerics<Integer>();
contenitore2.setObject(new Integer("1"));
Integer object2 = contenitore2.getObject(); //niente cast
System.out.println(object2);
```

pag. 381 - par. 11.3.1 - terza riga del paragrafo

il tipo generico era un `Integer`

Deve essere corretto in:

il tipo generico era `String`

pag. 386 - par. 11.3.4 – primo riquadro che riporta righe di codice

```
List<String> param = new ArrayList();
```

Deve essere corretto in:

```
List<String> param = new ArrayList<>();
```

pag. 392 - par. 11.4.2 – seconda riga del secondo riquadro che riporta righe di codice

```
public void mangia(Erbivoro erbivoro) {
```

Deve essere corretto in:

```
public void mangia(E erbivoro) {
```

pag. 411 - par. 12.2.4 - terza riga della pagina

Manca uno spazio tra il punto e l’inizio di una nuova frase:

della classe `Runtime`.Tuttavia l’utilizzo di questa

Deve essere corretto in:

della classe `Runtime`. Tuttavia l’utilizzo di questa

pag. 416 - par. 12.4.3 - prima riga del riquadro di console

```
java TestClassReflection Punto
```

Deve essere corretto in:

```
java TestClassReflection Object
```

pag. 417 - par. 12.4.4 – riga 9

```
javac –parameters NomeClasse.java
```

Deve essere corretto in:

```
javac –parameters Punto.java
```

pag. 431 - par. 12.6.1 - prime righe del secondo e del terzo riquadro di codice

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

Deve essere corretto in:

```
  ArrayList list = new ArrayList();
```

pag. 446 - par. 13.1 – riquadro di codice

C’è una parentesi di troppo:

```
      public void setAnni(int anni) {
          if (anni < 0)
              throw new IllegalArgumentException(anni
                + " anni, non è una età valida!";
              this.anni = anni;
          }
      }
```

Deve essere corretto in:

```
      public void setAnni(int anni) {
          if (anni < 0)
              throw new IllegalArgumentException(anni
                + " anni, non è una età valida!";
          this.anni = anni;
      }
```

pag. 466 - par. 13.3.2 - quartultima riga del secondo riquadro di codice

Ci sono delle virgolette superflue:

```
      @Deprecated (since="1.1", forRemoval="true")public void metodo() {
```

Deve essere corretto in:

```
      @Deprecated (since="1.1", forRemoval=true) public void metodo() {
```

pag. 475 - par. 14.1.1 - primo riquadro di codice

Il seguente listato:

```
  static {
      try {
          properties = new Properties();
          try {
              loadProperties();
              //...
          } catch (FileNotFoundException e) {
              //...
          }
      }
  }
```

Deve essere corretto in:

```
  static {
      properties = new Properties();
      try {
          loadProperties();
          //...
      } catch (FileNotFoundException e) {
          //...
      }
  }
```

pag. 475 - par. 14.1.1 - secondo riquadro di codice

La variabile `options` deve essere sostituita con `properties`, quindi il seguente listato:

```
  public static void loadProperties() throws FileNotFoundException {
      try (FileInputStream inputStream =
        new FileInputStream("resources/EJE_options.properties");) {
```

```
            options.load(inputStream);
            //...
        } catch (IOException e) {
            //...
        }
    }
```

Deve essere corretto in:

```
    public static void loadProperties() throws FileNotFoundException {
        try (FileInputStream inputStream =
          new FileInputStream("resources/EJE_options.properties");) {
            properties.load(inputStream);
            //...
        } catch (IOException e) {
            //...
        }
    }
```

pag. 486 - par. 14.1.4 - ultima linea del paragrafo

```
    System.out.printf("Data %d", new Date());
```

Deve essere corretto in:

```
    System.out.printf("Data %TD", new Date());
```

pag. 493 - par. 14.2.1 - descrizione del nome/prefisso with

Restituisce una copia dell’oggetto passato in input con un elemento cambiato.

Deve essere corretto in:

Restituisce una copia dell’oggetto su cui viene invocato, modificato con l’elemento passato in input.

pag. 527 - par. 15.3.4 - prima riga del paragrafo

Solo una percentuale ristretta di programmatori Java utilizza `Volatile`.

Deve essere corretto in:

Solo una percentuale ristretta di programmatori Java utilizza `volatile`.

pag. 544 - par. 15.6.1 - terza riga del riquadro di codice

```
public class ImmutableObject {
```

Deve essere corretto in:

```
public final class ImmutableObject {
```

pag. 553 - par. 15.6.3.5 - primo riquadro di codice

```
Festa festa = new Festa(cb);
```

Deve essere corretto in:

```
Festa festa = new Festa(luogoDellaFesta);
```

pag. 561 - par. 16.1.3.1 – terza riga della nota

Il ventesimo capitolo di questo libro invece, è dedicato alla creazione di interfacce grafiche con il nuovo standard JavaFX.

Deve essere corretto in:

L’appendice S di questo libro invece, è dedicata alla creazione di interfacce grafiche con JavaFX.

pag. 573 - par. 16.2.4 - seconda riga dopo il primo riquadro di codice

è possibile usare il nome il nome del tipo invece del nome di un oggetto.

Deve essere corretto in:

è possibile usare il nome del tipo invece del nome di un oggetto.

pag. 578 - par. 16.3.2 - primo riquadro di codice

Vi sono un paio di virgolette di troppo:

```
    film -> film.setNome("Star Wars Episodio 1" - La minaccia fantasma");
```

Deve essere corretto in:

```
    film -> film.setNome("Star Wars Episodio 1 - La minaccia fantasma");
```

pag. 593 - par. 17.2.2.2 - primo riquadro di codice

```
System.out.println(iterator.getName());
```

Deve essere corretto in:

```
System.out.println(iterator.getClass().getName());
```

pag. 609 – par. 17.7.1.1 – ottava e undicesima riga

I metodi setter e getter della variabile d’istanza nome, sono stati riportati erroneamente come `setTitolo()` e `getTitolo()`, ovvero alla riga otto:

```
public void setTitolo(String nome) {
```

Deve essere corretto in:

```
public void setNome(String nome) {
```

mentre alla riga undici:

```
public String getTitolo() {
```

Deve essere corretto in:

```
public String getNome() {
```

pag. 610 - par. 17.7.1.2 - primo riquadro di codice

C’è una lettera minuscola nella parola `arrayList` alla quinta riga, ovvero:

```
List<String> synchList = Collections.synchronizedList(arraylist);
```

Deve essere corretto in:

```
List<String> synchList = Collections.synchronizedList(arrayList);
```

pag. 613 - par. 17.7.3.2 - primo riquadro di codice

I generici di `Map` sono invertiti:

```
Map<String, Integer> immutableMap = new HashMap<Integer, String>();
```

Deve essere corretto in:

```
Map<Integer, String> immutableMap = new HashMap<Integer, String>();
```

pag. 613 - par. 17.7.3.2 - secondo riquadro di codice

```
Map immutableMap = Map.of(1, "a", 2, "b", 3, "c");
```

Deve essere corretto in:

```
Map<Integer, String> immutableMap = Map.of(1, "a", 2, "b", 3, "c");
```

pag. 614 - par. 17.7.3.2 - riquadro di console

```
jshell> Map immutableMap = Map.of(1, "a", 2, "b", 2, "c");
```

Deve essere corretto in:

```
jshell> Map<Integer, String> immutableMap = Map.of(1, "a", 2, "b", 2, "c");
```

pag. 614 - par. 17.7.3.3 - secondo riquadro di codice

```
List immutableList = List.of("a", "b", "c");
```

Deve essere corretto in:

```
List<String> immutableList = List.of("a", "b", "c");
```

pag. 641 - par. 18.3.3 - terzo e quarto punto del primo elenco puntato

Il terzo e quarto punto:

- `int read(char cbuf[], int offset, int length)`
- `throws IOException`

Devono essere corretti in:

- `int read(char cbuf[], int offset, int length) throws IOException`

pag. 641 - par. 18.3.3 - terzo e quarto punto del secondo elenco puntato

Il terzo e quarto punto:

- `int read(byte cbuf[], int offset, int length)`
- `throws IOException`

Devono essere corretti in:

- `int read(byte cbuf[], int offset, int length) throws IOException`

pag. 642 - par. 18.3.3 - terzo e quarto punto del primo elenco puntato

Il terzo e quarto punto:

- `int write(char cbuf[], int offset, int length)`
- `throws IOException`

Devono essere corretti in:

- `int write(char cbuf[], int offset, int length) throws IOException`

pag. 642 - par. 18.3.3 - terzo e quarto punto del secondo elenco puntato

Il terzo e quarto punto:

- `int write(byte cbuf[], int offset, int length)`
- `throws IOException`

Devono essere corretti in:

- `int write(byte cbuf[], int offset, int length) throws IOException`

pag. 656 - par. 18.5 – ultima riga della pagina

In particolare sono stati introdotti quattro nuovi package in Java 7: `java.nio.file`, `java.nio.file.spi`, `java.nio.file.attribute`, `java.nio.file.spi`.

Deve essere corretto in:

In particolare sono stati introdotti tre nuovi package in Java 7: `java.nio.file`, `java.nio.file.attribute`, `java.nio.file.spi`.

pag. 660 - par. 18.5.2 – sesta riga dopo il riquadro di codice

se il file `test.txt` è in sola lettura anche il file `test.txt` sarà in sola lettura

Deve essere corretto in:

se il file `test.txt` è in sola lettura anche il file `copy.txt` sarà in sola lettura

pag. 673 - par. 19.2.1 - terza riga dopo il primo riquadro di codice

```
C:/java9/capitolo19/hello-modular-world/src/cdsc.mymodule/java9/cap19
```

Deve essere corretto in:

```
C:/java9/capitolo_19/hello-modular-world/src/cdsc.mymodule/java9/cap19
```

pag. 673 - par. 19.2.1 - seconda riga dopo il secondo riquadro di codice

```
C:/java9/capitolo19/hello-modular-world
```

Deve essere corretto in:

```
C:/java9/capitolo_19/hello-modular-world
```

pag. 683 - par. 19.2.3.4 - prima riga del paragrafo

Le direttive `uses` e `provide to`

Deve essere corretto in:

Le direttive `uses` e `provides to`

pag. 686 - par. 19.3.2 – terza riga dopo il riquadro di codice

Le parole sono prive di spazi separatori, quindi:

Sinoticheaquestaapplicazionemancaunaclasseconil`main()`peressereeseguita.Atalepropositoabbiamoimplementatounmoduloaparte

Deve essere corretto in:

Si noti che a questa applicazione manca una classe con il `main()` per essere eseguita. A tale proposito abbiamo implementato un modulo a parte

pag. 686 - par. 19.3.2 – quinta riga dopo il riquadro di codice

non riportato nel precedente programma

Deve essere corretto in:

non riportato nel precedente diagramma

pag. 688 - par. 19.3.3 - prima riga dopo il secondo riquadro di codice

Mentre il descrittore del modulo `com.claudiodesio.inves`

Deve essere corretto in:

Mentre il descrittore del modulo `com.claudiodesio.invs`

L’autore vuole ringraziare tutte le persone che hanno contribuito a realizzare questa errata corrige. In particolare Ettore Gallina (https://www.gallinaettore.com) che ha contribuito con un gran numero di segnalazioni, mostrando grande competenza e attenzione. Grazie davvero!

Claudio De Sio Cesari