



Oggetti e Java

Capitolo 3 ottobre 2011

Contenuti

- ◆ Il linguaggio di programmazione Java
- ◆ Introduzione alla programmazione in Java
 - classi Java
 - programmi Java
 - API di Java
 - programmazione in Java
- ◆ Introduzione a Java (per esempi)
 - scrittore sullo schermo
 - calcolo di una radice quadrata
 - perimetro di un triangolo
 - robot in un labirinto
 - lettura e somma di due numeri interi
 - una classe per istanziare oggetti

Oggetti e Java

Java è un linguaggio di programmazione a oggetti

- questo capitolo introduce la programmazione in Java
- concretizzando gli esempi visti nel precedente capitolo

Introduzione alla programmazione in Java

Java

- è un linguaggio di programmazione orientato agli oggetti
- sviluppato dalla Sun Microsystems, e rilasciato nel 1995
- indipendente dalla piattaforma
 - write once, run everywhere
 - realizzato per le reti di calcolatori e supportato dai principali browser Web
- sito web **java.sun.com**



- la Sun è stata acquisita dalla Oracle nel 2009
 - sito web **www.oracle.com/technetwork/java**

Introduzione alla programmazione in Java

Java è un linguaggio di programmazione

- permette di scrivere programmi

Java è un linguaggio di programmazione orientato agli oggetti

- permette di definire classi

Programmi Java

- diversi tipi di programmi java: applicazioni, applet e servlet
- un'applicazione Java
 - eseguita da una macchina virtuale Java (JVM) autonoma
 - definita da classe applicazione, con il metodo speciale **main**
 - possibile la presenza di ulteriori classi

Introduzione alla programmazione in Java

API di Java

- librerie di oggetti e classi predefinite, organizzate in package
 - una libreria standard, parte di Java SDK
 - librerie aggiuntive – ad esempio, il package **fi**ji

Programmazione in Java

- sintassi e semantica
- uso di oggetti e classi predefiniti – dalle API di Java utilizzate
- definizione di nuove classi

Introduzione a Java (per esempi)

Inizia ora una panoramica sulla programmazione in Java

- per ora, l'enfasi è su come **leggere** alcuni programmi Java – di complessità via via crescente
- alla **progettazione** e **scrittura** di programmi Java sono dedicati molti dei successivi capitoli

Scrittore sullo schermo

Si vuole scrivere un'applicazione Java che visualizza sullo schermo le seguenti frasi

```
ciao a tutti
questo testo introduce
i fondamenti dell'informatica
```

In generale, per scrivere un'applicazione è necessario definire una classe che è, appunto, il progetto di un'applicazione Java

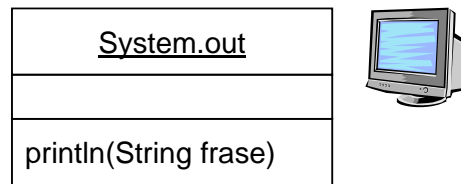
- un'applicazione Java è un oggetto classe che sa eseguire un'operazione speciale **main**

Inoltre, l'applicazione deve avere l'effetto desiderato

- l'effetto dell'esecuzione dell'applicazione coincide con quello dell'operazione speciale **main** – che pertanto deve essere quello di visualizzare le tre frasi sullo schermo

System.out

L'applicazione ha bisogno di visualizzare delle frasi sullo schermo



- **System.out** è un oggetto definito dalle API di Java
- **System.out** modella lo schermo del calcolatore
- **System.out** sa eseguire una operazione **println** che visualizza una frase (il parametro dell'operazione)

L'applicazione ScrittoreSulloSchermo

```
/* Applicazione che visualizza una frase sullo schermo. */  
public class ScrittoreSulloSchermo {  
    public static void main(String[] args) {  
        System.out.println("ciao a tutti");  
        System.out.println("questo testo introduce");  
        System.out.println("i fondamenti dell'informatica");  
    }  
}
```

Definizione di una classe

```
/* Applicazione che visualizza una frase sullo schermo. */  
public class ScrittoreSulloSchermo {  
    public static void main(String[] args) {  
        System.out.println("ciao a tutti");  
        System.out.println("questo testo introduce");  
        System.out.println("i fondamenti dell'informatica");  
    }  
}
```

- intestazione – nome della classe preceduto da **public class**
- corpo
- modalità di costruzione statica
- oggetto classe **ScrittoreSulloSchermo**

Un commento

```
/* Applicazione che visualizza una frase sullo schermo. */  
public class ScrittoreSulloSchermo {  
    public static void main(String[] args) {  
        System.out.println("ciao a tutti");  
        System.out.println("questo testo introduce");  
        System.out.println("i fondamenti dell'informatica");  
    }  
}
```

Definizione di un metodo

```
/* Applicazione che visualizza una frase sullo schermo. */
public class ScrittoreSulloSchermo {
    public static void main(String[] args) {
        System.out.println("ciao a tutti");
        System.out.println("questo testo introduce");
        System.out.println("i fondamenti dell'informatica");
    }
}
```

- nel corpo della classe
- un metodo implementa un'operazione
- intestazione – prototipo
- corpo
- **main**
- **ScrittoreSulloSchermo** è una classe applicazione

Una sequenza di istruzioni

```
/* Applicazione che visualizza una frase sullo schermo. */
public class ScrittoreSulloSchermo {
    public static void main(String[] args) {
        System.out.println("ciao a tutti");
        System.out.println("questo testo introduce");
        System.out.println("i fondamenti dell'informatica");
    }
}
```

- nel corpo del metodo
- ogni **istruzione** descrive un'**azione**
- le azioni descritte da queste istruzioni vengono eseguite se e quando qualcuno chiede a **ScrittoreSulloSchermo** di eseguire l'operazione **main**

Istruzioni

```
/* Applicazione che visualizza una frase sullo schermo. */
public class ScrittoreSulloSchermo {
    public static void main(String[] args) {
        System.out.println("ciao a tutti");
        System.out.println("questo testo introduce");
        System.out.println("i fondamenti dell'informatica");
    }
}
```

- questa istruzione descrive l'invio di un messaggio a un oggetto
 - mittente: ...
 - messaggio: **println("ciao a tutti")**
 - destinatario: **System.out**
- punti, parentesi, virgolette, punti e virgola, graffe, ...

Istruzioni e azioni

```
/* Applicazione che visualizza una frase sullo schermo. */
public class ScrittoreSulloSchermo {
    public static void main(String[] args) {
        System.out.println("ciao a tutti");
        System.out.println("questo testo introduce");
        System.out.println("i fondamenti dell'informatica");
    }
}
```

- effetto dell'esecuzione di questa istruzione
 - **ScrittoreSulloSchermo**
 - invia il messaggio **println("ciao a tutti")**
 - all'oggetto **System.out**
- l'effetto "visibile" dell'esecuzione di questa istruzione
 - **System.out** visualizza (su se stesso) la stringa *ciao a tutti*

Esecuzione di un metodo

```
/* Applicazione che visualizza una frase sullo schermo. */
public class ScrittoreSulloSchermo {
    public static void main(String[] args) {
        System.out.println("ciao a tutti");
        System.out.println("questo testo introduce");
        System.out.println("i fondamenti dell'informatica");
    }
}
```

- per **ScrittoreSulloSchermo** eseguire l'operazione **main** vuol dire eseguire una alla volta e in sequenza le istruzioni scritte nel corpo del metodo **main**

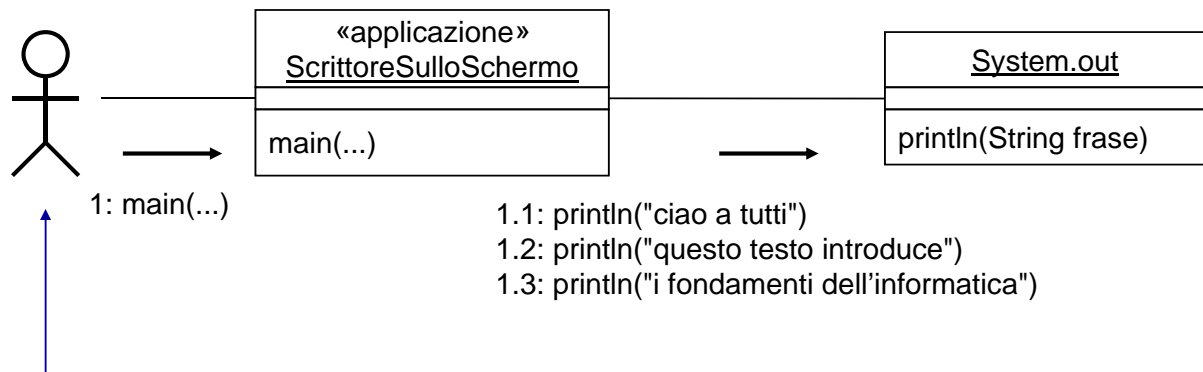
Esecuzione di un'applicazione

```
/* Applicazione che visualizza una frase sullo schermo. */
public class ScrittoreSulloSchermo {
    public static void main(String[] args) {
        System.out.println("ciao a tutti");
        System.out.println("questo testo introduce");
        System.out.println("i fondamenti dell'informatica");
    }
}
```

- l'esecuzione dell'applicazione **ScrittoreSulloSchermo** consiste nell'esecuzione dell'operazione **main** da parte dell'oggetto classe **ScrittoreSulloSchermo**

```
ciao a tutti
questo testo introduce
i fondamenti dell'informatica
```

Diagramma di collaborazione per ScrittoreSulloSchermo



questa icona rappresenta
l'utente del programma
(tramite la JVM)

Esercizio

Commentare la definizione della seguente classe

```
/* Applicazione che visualizza sullo schermo
 * la poesia Mattino di Giuseppe Ungaretti. */
public class Mattino {
    public static void main(String[] args) {
        System.out.println("M'illumino");
        System.out.println("d'immenso");
    }
}
```

Esercizio

Una lunga tradizione vuole che il primo programma scritto da un programmatore sia quello che visualizza sullo schermo la frase *Hello, world* (che significa *Ciao, mondo*)

- definire un'applicazione Java **HelloWorld** che visualizza sullo schermo questa frase

Calcolo di una radice quadrata

Si vuole scrivere un'applicazione Java che calcola e visualizza la radice quadrata di 144

- l'esecuzione di questo programma dovrà visualizzare sullo schermo

12

<u>Math</u>
double sqrt(double n)



<u>System.out</u>
println(double x)



Math, come **System.out**, è un oggetto delle API di Java

L'applicazione RadiceQuadrata

```
/* Applicazione che calcola e visualizza sullo schermo
 * la radice quadrata di 144. */
public class RadiceQuadrata {
    public static void main(String[] args) {
        double radice;

        radice = Math.sqrt(144);
        System.out.println(radice);
    }
}
```

Che cosa sappiamo dire di questa classe? Che cosa non riusciamo ancora a comprendere?

Variabili e dichiarazioni di variabili

```
/* Applicazione che calcola e visualizza sullo schermo
 * la radice quadrata di 144. */
public class RadiceQuadrata {
    public static void main(String[] args) {
        double radice;

        radice = Math.sqrt(144);
        System.out.println(radice);
    }
}
```

- **radice** è una variabile
- in Java, **double** è il tipo del numeri reali
 - tipo = dominio di valori + operazioni
- qual'è lo scopo di **radice**? memorizzare la radice quadrata di 144

Invio di un messaggio a un oggetto

```
/* Applicazione che calcola e visualizza sullo schermo
 * la radice quadrata di 144. */
public class RadiceQuadrata {
    public static void main(String[] args) {
        double radice;

        radice = Math.sqrt(144);
        System.out.println(radice);
    }
}
```

Istruzione di assegnazione

```
/* Applicazione che calcola e visualizza sullo schermo
 * la radice quadrata di 144. */
public class RadiceQuadrata {
    public static void main(String[] args) {
        double radice;

        radice = Math.sqrt(144);
        System.out.println(radice);
    }
}
```

- un'assegnazione
 - valuta un'espressione – **Math.sqrt(144)**
 - memorizza il valore calcolato in una variabile – **radice**

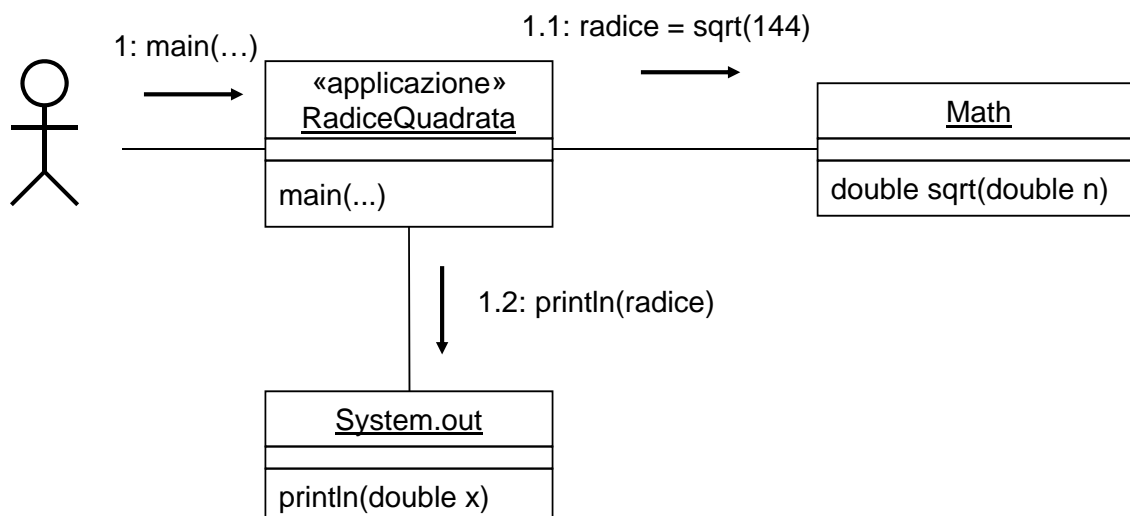
Visualizzazione di un valore

```
/* Applicazione che calcola e visualizza sullo schermo
 * la radice quadrata di 144. */
public class RadiceQuadrata {
    public static void main(String[] args) {
        double radice;

        radice = Math.sqrt(144);
        System.out.println(radice);
    }
}
```

- questa istruzione ha lo scopo di visualizzare il valore di **radice**
 - accesso al valore di una variabile vs. memorizzazione di un valore in una variabile
 - scopo delle virgolette

Diagramma di collaborazione per RadiceQuadrata



Esercizio – commentare la seguente classe

```
/* Applicazione che calcola e visualizza
 * l'ipotenusa di un triangolo rettangolo. */
public class Ipotenusa {
    public static void main(String[] args) {
        /* i due cateti */
        double cateto1, cateto2;
        /* somma dei quadrati dei cateti */
        double sommaQuadrati;
        /* ipotenusa del triangolo, da calcolare */
        double ipotenusa;

        /* assegna un valore ai due cateti */
        cateto1 = 3;
        cateto2 = 4;
        /* calcola l'ipotenusa */
        sommaQuadrati = cateto1*cateto1 + cateto2*cateto2;
        ipotenusa = Math.sqrt(sommaQuadrati);
        /* visualizza l'ipotenusa */
        System.out.println(ipotenusa);
    }
}
```

29

Espressioni aritmetiche

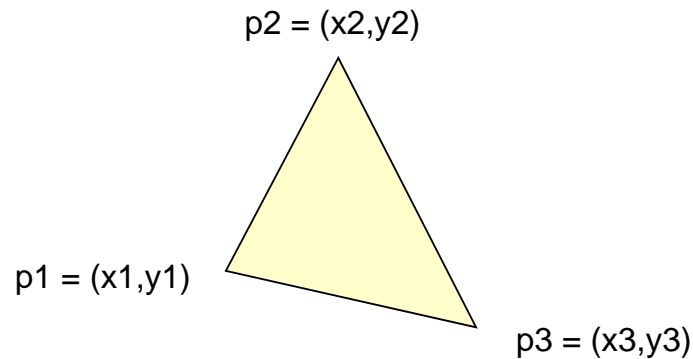
```
/* Applicazione che calcola e visualizza
 * l'ipotenusa di un triangolo rettangolo. */
public class Ipotenusa {
    public static void main(String[] args) {
        /* i due cateti */
        double cateto1, cateto2;
        /* somma dei quadrati dei cateti */
        double sommaQuadrati;
        /* ipotenusa del triangolo, da calcolare */
        double ipotenusa;

        /* assegna un valore ai due cateti */
        cateto1 = 3;
        cateto2 = 4;
        /* calcola l'ipotenusa */
        sommaQuadrati = cateto1*cateto1 + cateto2*cateto2;
        ipotenusa = Math.sqrt(sommaQuadrati);
        /* visualizza l'ipotenusa */
        System.out.println(ipotenusa);
    }
}
```

30

Perimetro di un triangolo

Si vuole scrivere un'applicazione che calcola e visualizza il perimetro di un triangolo di cui sono noti i vertici



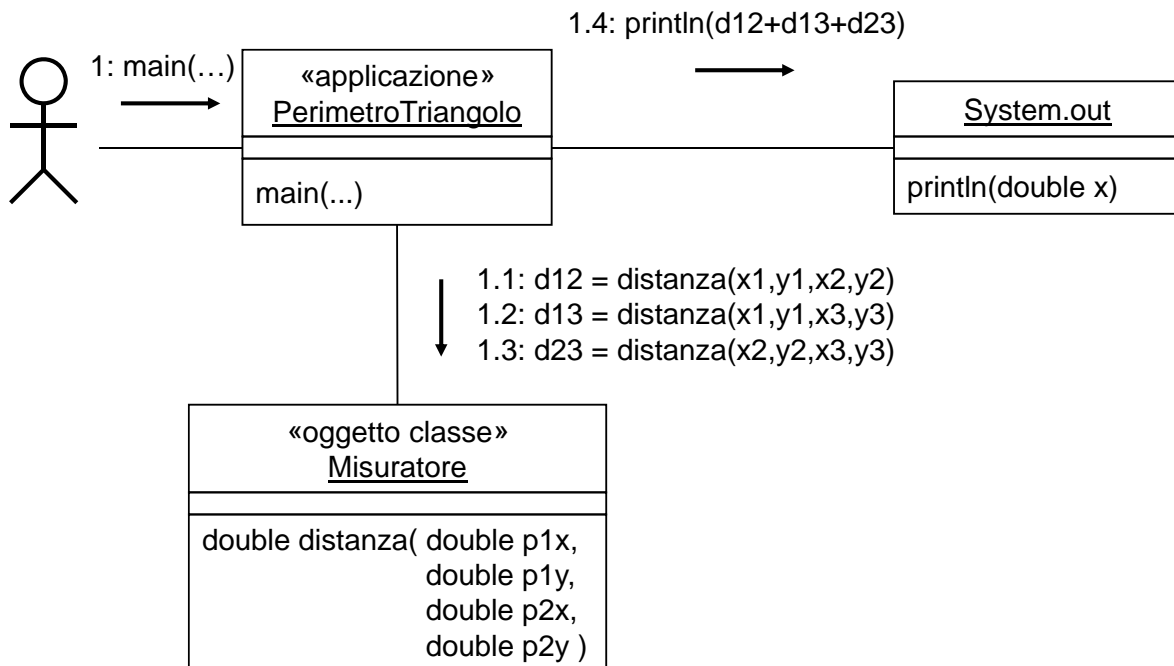
L'oggetto Misuratore

Useremo l'oggetto classe **Misuratore**

- sa calcolare la distanza tra due punti, date le coordinate dei punti

«oggetto classe» <u>Misuratore</u>
double distanza(double p1x, double p1y, double p2x, double p2y)

Collaborazione che si vuole realizzare



L'applicazione PerimetroTriangolo

```
/* Applicazione che calcola e visualizza
 * il perimetro di un triangolo. */
public class PerimetroTriangolo {
    public static void main(String[] args) {
        double x1, y1;           // coordinate primo vertice
        double x2, y2;           // coordinate secondo vertice
        double x3, y3;           // coordinate terzo vertice
        double d12, d13, d23;    // distanze tra i vertici
        double perimetro;        // perimetro del triangolo

        /* imposta le coordinate dei vertici */
        x1 = 1;
        y1 = 2;
        x2 = 4;
        y2 = 6;
        x3 = 8;
        y3 = 1;
```

... segue ...

un altro formato per i commenti,
da `//` alla fine della linea

L'applicazione PerimetroTriangolo

... segue ...

```
/* calcola le distanze tra i vertici */
d12 = Misuratore.distanza(x1, y1, x2, y2);
d13 = Misuratore.distanza(x1, y1, x3, y3);
d23 = Misuratore.distanza(x2, y2, x3, y3);

/* calcola il perimetro del triangolo */
perimetro = d12 + d13 + d23;

/* visualizza il perimetro */
System.out.print("Il perimetro del triangolo è ");
System.out.println(perimetro);
}
}
```

System.out sa eseguire anche una operazione **print**,
che visualizza l'argomento, senza poi andare a capo
(come fa **println**)

L'oggetto classe Misuratore

L'oggetto classe **Misuratore**

- non fa parte delle API di Java
- non è un oggetto predefinito

Bisogna allora definire anche l'oggetto **Misuratore**

- bisogna definire una classe per l'oggetto **Misuratore**
- questo oggetto deve saper eseguire l'operazione **distanza**

La definizione dell'applicazione **PerimetroTriangolo** richiede la definizione di due classi

La classe Misuratore

```
/* Oggetto che sa calcolare la distanza tra due punti. */
public class Misuratore {
    /* Calcola la distanza tra i punti p1 e p2
     * di coordinate (p1x,p1y) e (p2x,p2y) */
    public static double distanza(double p1x, double p1y,
                                   double p2x, double p2y) {
        double qd;    // quadrato della distanza tra p1 e p2
        double d;     // distanza tra p1 e p2

        /* calcola il quadrato della distanza tra p1 e p2 */
        qd = (p1x-p2x)*(p1x-p2x) + (p1y-p2y)*(p1y-p2y);

        /* calcola la distanza tra p1 e p2 */
        d = Math.sqrt(qd);

        /* restituisce la distanza tra p1 e p2 */
        return d;
    }
}
```

La classe Misuratore e l'oggetto classe Misuratore

```
/* Oggetto che sa calcolare la distanza tra due punti. */
public class Misuratore {
    ...
}
```

- **Misuratore** è una classe a modalità di costruzione statica
 - dalla classe **Misuratore** viene costruito un oggetto classe
 - questo oggetto classe si chiama **Misuratore**

Misuratore e il metodo distanza

```
/* Oggetto che sa calcolare la distanza tra due punti. */
public class Misuratore {
    /* Calcola la distanza tra i punti p1 e p2
    * di coordinate (p1x,p1y) e (p2x,p2y) */
    public static double distanza(double p1x, double p1y,
                                  double p2x, double p2y) {
        ...
    }
}
```

- il corpo della classe **Misuratore** descrive le caratteristiche dell'oggetto **Misuratore**
 - ogni metodo implementa un'operazione
 - l'oggetto **Misuratore** sa eseguire l'operazione **distanza**

Prototipo e parametri del metodo distanza

```
/* Calcola la distanza tra i punti p1 e p2
* di coordinate (p1x,p1y) e (p2x,p2y) */
public static double distanza(double p1x, double p1y,
                              double p2x, double p2y) {
    double qd;    // quadrato della distanza tra p1 e p2
    double d;    // distanza tra p1 e p2

    /* calcola il quadrato della distanza tra p1 e p2 */
    qd = (p1x-p2x)*(p1x-p2x) + (p1y-p2y)*(p1y-p2y);

    /* calcola la distanza tra p1 e p2 */
    d = Math.sqrt(qd);

    /* restituisce la distanza tra p1 e p2 */
    return d;
}
```

Parametri formali e parametri attuali

Definizione del metodo **distanza**

```
public static double distanza(double p1x, double p1y,  
                             double p2x, double p2y)
```

Uso del metodo **distanza**

```
d12 = Misuratore.distanza(1, 2, 4, 6);
```

Per richiedere l'esecuzione di un metodo è necessario specificare un valore per ciascuno dei parametri del metodo

- **parametri formali** del metodo
- **parametri attuali** dell'invocazione del metodo
- **legame dei parametri**

Esecuzione del metodo **distanza**

```
/* Calcola la distanza tra i punti p1 e p2  
 * di coordinate (p1x,p1y) e (p2x,p2y) */  
public static double distanza(double p1x, double p1y,  
                             double p2x, double p2y) {  
    double qd;    // quadrato della distanza tra p1 e p2  
    double d;    // distanza tra p1 e p2  
  
    /* calcola il quadrato della distanza tra p1 e p2 */  
    qd = (p1x-p2x)*(p1x-p2x) + (p1y-p2y)*(p1y-p2y);  
  
    /* calcola la distanza tra p1 e p2 */  
    d = Math.sqrt(qd);  
  
    /* restituisce la distanza tra p1 e p2 */  
    return d;  
}
```

- **legame dei parametri**
- **esecuzione delle istruzioni nel corpo**
- **restituzione di un valore**

Restituzione di un valore

Definizione del metodo distanza

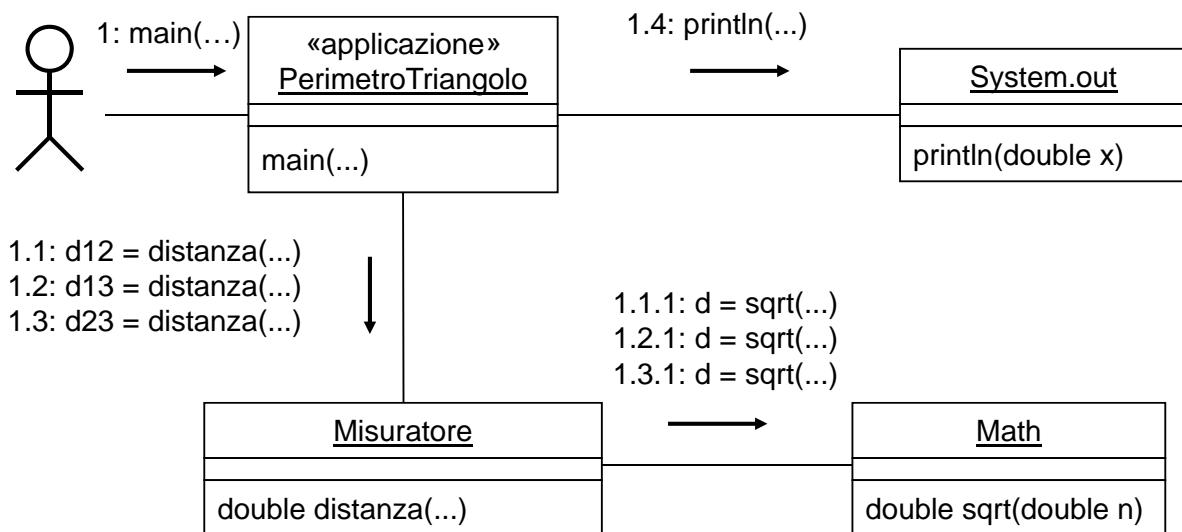
```
public static double distanza(double p1x, double p1y,  
                               double p2x, double p2y) {  
    ...  
    return d;  
}
```

Uso del metodo distanza

```
d12 = Misuratore.distanza(1, 2, 4, 6);
```

- quando un metodo restituisce un valore, solitamente la richiesta dell'esecuzione del metodo viene fatta nell'ambito di un'assegnazione

Diagramma di collaborazione per PerimetroTriangolo



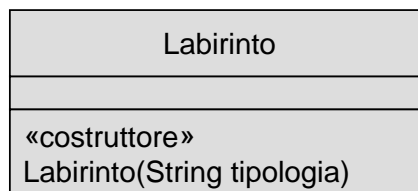
Robot in un labirinto

Si vuole scrivere un'applicazione Java che fa attraversare un labirinto a un robot

- esempio complesso, introdotto gradualmente
- inizialmente, si vuole scrivere un'applicazione **CreazioneLabirinto** che crea e visualizza un labirinto

La classe Labirinto

La classe **Labirinto** del package **fiji.robot** permette di istanziare oggetti che modellano labirinti che sanno visualizzarsi sullo schermo



Ma come si istanzia un oggetto da una classe?

L'applicazione CreazioneLabirinto

```
import fiji.robot.*;
/* Applicazione che crea un labirinto semplice. */
public class CreazioneLabirinto {
    public static void main(String[] args) {
        Labirinto l;

        l = new Labirinto("semplice");
    }
}
```

Creazione di un oggetto Labirinto

```
import fiji.robot.*;
/* Applicazione che crea un labirinto semplice. */
public class CreazioneLabirinto {
    public static void main(String[] args) {
        Labirinto l;

        l = new Labirinto("semplice");
    }
}
```

Memorizzazione del riferimento al Labirinto creato

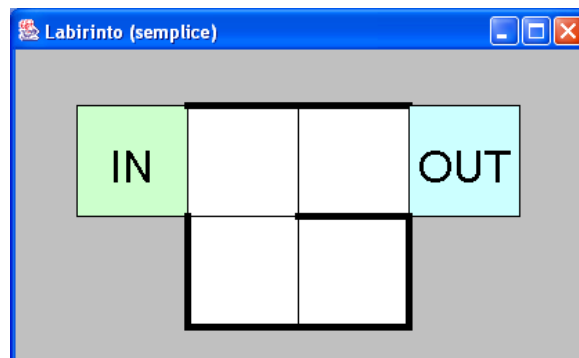
```
import fiji.robot.*;
/* Applicazione che crea un labirinto semplice. */
public class CreazioneLabirinto {
    public static void main(String[] args) {
        Labirinto l;

        l = new Labirinto("semplice");
    }
}
```

Esecuzione di CreazioneLabirinto

```
import fiji.robot.*;
/* Applicazione che crea un labirinto semplice. */
public class CreazioneLabirinto {
    public static void main(String[] args) {
        Labirinto l;

        l = new Labirinto("semplice");
    }
}
```



La clausola import

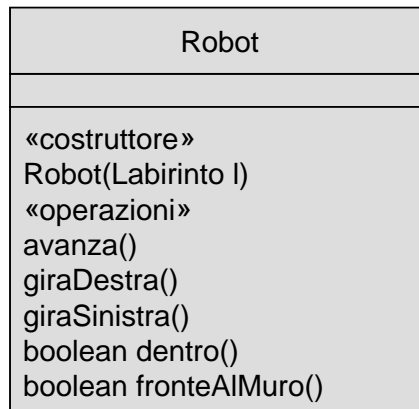
```
import fiji.robot.*;
/* Applicazione che crea un labirinto semplice. */
public class CreazioneLabirinto {
    public static void main(String[] args) {
        Labirinto l;

        l = new Labirinto("semplice");
    }
}
```

Creazione di un robot in un labirinto

L'applicazione **RobotInLabirinto** deve creare e visualizzare un robot in un labirinto

- la classe **Robot** del package **fiji.robot** permette di istanziare oggetti che modellano robot

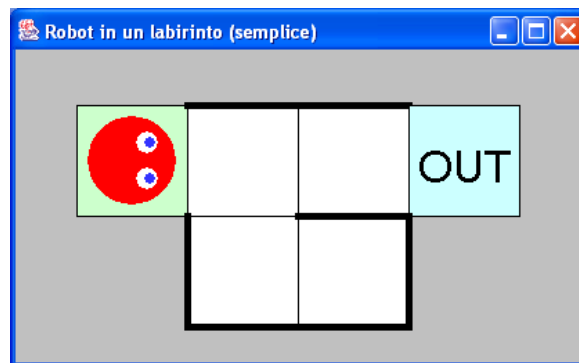


- il costruttore di **Robot** vuole come argomento un **Labirinto**
 - quello in cui il robot deve essere collocato, nella cella d'ingresso e nella direzione d'ingresso

L'applicazione RobotInLabirinto

```
import fiji.robot.*;
/* Applicazione che crea un robot in un labirinto. */
public class RobotInLabirinto {
    public static void main(String[] args) {
        Labirinto l;
        Robot r;

        /* crea il labirinto l */
        l = new Labirinto("semplice");
        /* crea il robot r nel labirinto l */
        r = new Robot(l);
    }
}
```



53

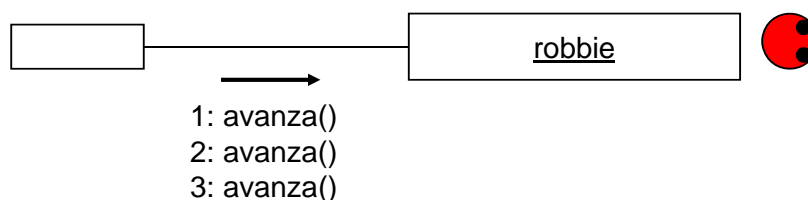
Oggetti e Java

Fondamenti di informatica: Oggetti e Java
Luca Cabibbo

Attraversamento di un labirinto semplice

L'applicazione **AttraversamentoLabirintoSemplice** deve

- crea un labirinto semplice
- crea un robot nel labirinto semplice
- fa attraversare il labirinto semplice al robot, facendolo avanzare per tre volte



- infatti, gli oggetti **Robot** sanno eseguire l'operazione **avanza()**

54

Oggetti e Java

Fondamenti di informatica: Oggetti e Java
Luca Cabibbo

L'applicazione AttraversamentoLabirintoSemplice

```
import fiji.robot.*;
/* Attraversamento di un labirinto semplice. */
public class AttraversamentoLabirintoSemplice {
    public static void main(String[] args) {
        Labirinto l;
        Robot r;

        /* crea il labirinto l */
        l = new Labirinto("semplice");
        /* crea il robot r nel labirinto l */
        r = new Robot(l);

        /* fa attraversare a r il labirinto */
        r.avanza();
        r.avanza();
        r.avanza();
    }
}
```

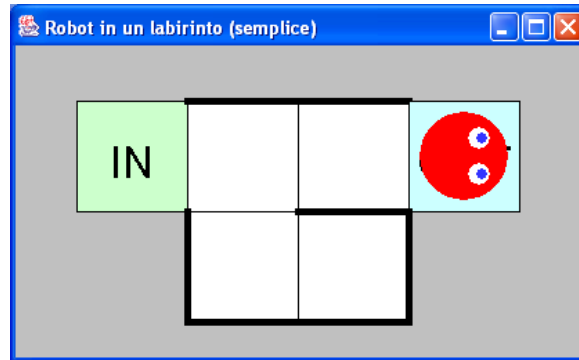
L'applicazione AttraversamentoLabirintoSemplice

```
/* fa attraversare a r il labirinto */
r.avanza();
r.avanza();
r.avanza();
```

- l'espressione **r.avanza()** ha lo scopo di richiedere all'oggetto il cui riferimento è memorizzato nella variabile **r** di eseguire l'operazione **avanza()**

Esecuzione di AttraversamentoLabirintoSemplice

```
/* crea il labirinto l */  
l = new Labirinto("semplice");  
/* crea il robot r nel labirinto l */  
r = new Robot(l);  
  
/* fa attraversare a r il labirinto */  
r.avanza();  
r.avanza();  
r.avanza();
```

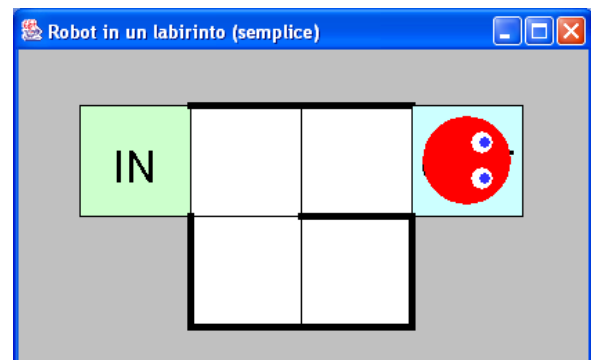
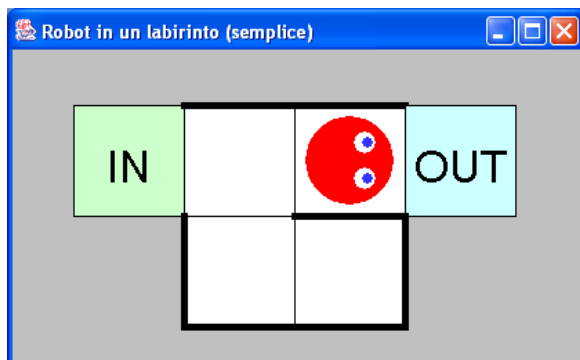
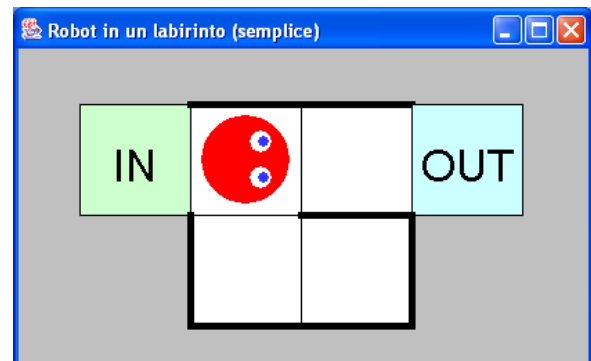
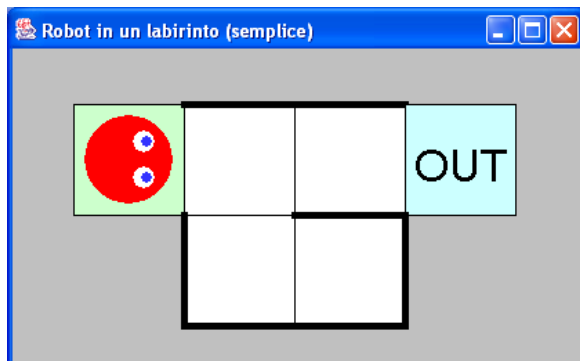


57

Oggetti e Java

Fondamenti di informatica: Oggetti e Java
Luca Cabibbo

Esecuzione di AttraversamentoLabirintoSemplice



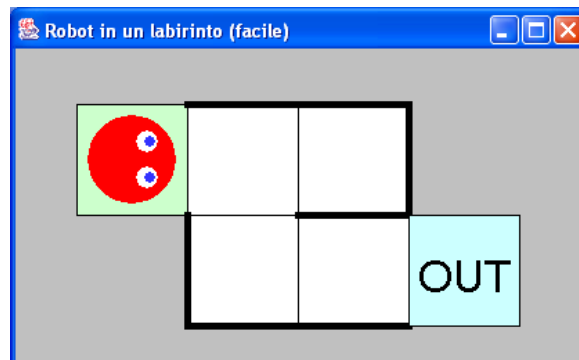
58

Oggetti e Java

Fondamenti di informatica: Oggetti e Java
Luca Cabibbo

Esercizio

Definire l'applicazione **AttraversamentoLabirintoFacile**, per far attraversare a un robot un labirinto "facile"



- usare anche le operazioni **giraDestra()** e **giraSinistra()**

Conclusioni

Per usare un oggetto di tipo **Robot** – ignorando l'uso di **Labirinto**, che è simile – è stato (ed è) necessario

- importare la classe **Robot** dal package **fiji.robot** – usando la clausola **import fiji.robot.*;**
- dichiarare una variabile **r** di tipo **Robot**
- creare l'oggetto che rappresenta il robot mediante un'istruzione **r = new Robot(l);** – in cui **l** è il riferimento ad un **Labirinto**
- usare opportunamente l'oggetto **r** mediante le operazioni di **Robot**
 - **avanza()**
 - **giraDestra()**
 - **giraSinistra()**
 - **boolean dentro()**
 - **boolean fronteAIMuro()**

Lettura e somma di due numeri interi




Si vuole scrivere un'applicazione Java che legge dalla tastiera due numeri interi, ne calcola la somma e la visualizza sullo schermo

```
Scrivi due numeri interi
10 15
La somma dei due numeri è 25
```


Un oggetto che rappresenta la tastiera del calcolatore



Le API di Java hanno un oggetto **System.in** che rappresenta la tastiera del calcolatore, ma che non è semplice da usare

<u>System.in</u> : InputStream	
int read()	

In questo testo viene usato l'oggetto **Letttore.in**, del package **fiji.io**

<u>Letttore.in</u>	
char leggiChar() int leggiInt() double leggiDouble() String leggiLinea() boolean eoln()	

L'applicazione SommaDueNumeri



```
import fiji.io.*;
/* Applicazione che legge dalla tastiera due numeri interi
 * e ne calcola e visualizza la somma. */
public class SommaDueNumeri {
    public static void main(String[] args) {
        int a;           // il primo numero intero
        int b;           // il secondo numero intero
        int somma;       // la somma di a e b

        /* legge i due numeri interi a e b */
        System.out.println("Scrivi due numeri interi");
        /* legge due numeri interi a e b */
        a = Lettore.in.leggiInt();
        b = Lettore.in.leggiInt();
        /* calcola la somma di a e b e la visualizza */
        somma = a+b;
        System.out.print("La somma dei due numeri è ");
        System.out.println(somma);
    }
}
```

Che cosa c'è di nuovo?



```
import fiji.io.*;
/* Applicazione che legge dalla tastiera due numeri interi
 * e ne calcola e visualizza la somma. */
public class SommaDueNumeri {
    public static void main(String[] args) {
        int a;           // il primo numero intero
        int b;           // il secondo numero intero
        int somma;       // la somma di a e b

        /* legge i due numeri interi a e b */
        System.out.println("Scrivi due numeri interi");
        /* legge due numeri interi a e b */
        a = Lettore.in.leggiInt();
        b = Lettore.in.leggiInt();
        /* calcola la somma di a e b e la visualizza */
        somma = a+b;
        System.out.print("La somma dei due numeri è ");
        System.out.println(somma);
    }
}
```

Esercizio



Scrivere un'applicazione che legge dalla tastiera un numero razionale, ne calcola la radice quadrata e la visualizza sullo schermo

```
Scrivi un numero
```

```
1.21
```

```
La radice quadrata di 1.21 è 1.1
```

- per leggere dalla tastiera un numero razionale bisogna usare il metodo **double leggiDouble()** di **Letto.re.in**

Lettura e somma di due numeri interi

Si vuole scrivere un'applicazione Java che legge dalla tastiera due numeri interi, ne calcola la somma e la visualizza sullo schermo


```
Scrivi due numeri interi
```

```
10 15
```

```
La somma dei due numeri è 25
```

Un oggetto che rappresenta la tastiera del calcolatore

Le API di Java hanno un oggetto **System.in** che rappresenta la tastiera del calcolatore, ma che non è semplice da usare direttamente


<u>System.in : InputStream</u>	
int read()	

In questo corso faremo però riferimento ad un altro oggetto delle API di Java che può essere usato come rappresentante della tastiera

- l'oggetto **in** di tipo **Scanner**, del package **java.util**

Un oggetto che rappresenta la tastiera del calcolatore

L'oggetto **in** di tipo **Scanner**, del package **java.util**

<u>in : Scanner</u>	
int nextInt() double nextDouble() String nextLine() String next() boolean hasNextInt() boolean hasNextDouble() boolean hasNextLine() boolean hasNext() ...	

Uso di Scanner

Per usare un oggetto di tipo **Scanner** bisogna

- importare la classe **Scanner** dal package **java.util** usando la clausola **import java.util.Scanner;** oppure **import java.util.*;**
- dichiarare una variabile **in** di tipo **Scanner**
- creare l'oggetto che rappresenta la tastiera mediante un'istruzione **in = new Scanner(System.in);**
- usare opportunamente le operazioni dell'oggetto **in**
 - **int nextInt()**
 - **double nextDouble()**
 - **String nextLine()**
 - **String next()**
 - **boolean hasNextInt()**
 - **boolean hasNextDouble()**
 - **boolean hasNextLine()**
 - **boolean hasNext()**
 - ...

L'applicazione SommaDueNumeri

```
import java.util.*;
/* Applicazione che legge dalla tastiera due numeri interi
 * e ne calcola e visualizza la somma. */
public class SommaDueNumeri {
    public static void main(String[] args) {
        int a;           // il primo numero intero
        int b;           // il secondo numero intero
        int somma;       // la somma di a e b
        Scanner in;      // per la lettura dalla tastiera

        /* crea l'oggetto che rappresenta la tastiera */
        in = new Scanner( System.in );

        /* legge i due numeri interi a e b */
        System.out.println("Scrivi due numeri interi");
        /* legge due numeri interi a e b */
        a = in.nextInt();
        b = in.nextInt();
        /* calcola la somma di a e b e la visualizza */
        somma = a+b;
        System.out.print("La somma dei due numeri è ");
        System.out.println(somma);
    }
}
```

Che cosa c'è di nuovo?

```
import java.util.*;
/* Applicazione che legge dalla tastiera due numeri interi
 * e ne calcola e visualizza la somma. */
public class SommaDueNumeri {
    public static void main(String[] args) {
        int a;           // il primo numero intero
        int b;           // il secondo numero intero
        int somma;       // la somma di a e b
        Scanner in;      // per la lettura dalla tastiera

        /* crea l'oggetto che rappresenta la tastiera */
        in = new Scanner( System.in );

        /* legge i due numeri interi a e b */
        System.out.println("Scrivi due numeri interi");
        /* legge due numeri interi a e b */
        a = in.nextInt();
        b = in.nextInt();
        /* calcola la somma di a e b e la visualizza */
        somma = a+b;
        System.out.print("La somma dei due numeri è ");
        System.out.println(somma);
    }
}
```

71

Stessa applicazione, con Lettore.in

```
import fiji.io.*;
/* Applicazione che legge dalla tastiera due numeri interi
 * e ne calcola e visualizza la somma. */
public class SommaDueNumeri {
    public static void main(String[] args) {
        int a;           // il primo numero intero
        int b;           // il secondo numero intero
        int somma;       // la somma di a e b

        /* legge i due numeri interi a e b */
        System.out.println("Scrivi due numeri interi");
        /* legge due numeri interi a e b */
        a = Lettore.in.leggiInt();
        b = Lettore.in.leggiInt();
        /* calcola la somma di a e b e la visualizza */
        somma = a+b;
        System.out.print("La somma dei due numeri è ");
        System.out.println(somma);
    }
}
```

72

Confronto

```
import java.util.*;
/* Applicazione che legge dalla tastiera due numeri interi
 * e ne calcola e visualizza la somma. */
public class SommaDueNumeri {
    public static void main(String[] args) {
        int a;        // il primo numero intero
        int b;        // il secondo numero intero
        int somma;    // la somma di a e b
        Scanner in;   // per la lettura dalla tastiera

        /* crea l'oggetto che rappresenta la tastiera */
        in = new Scanner( System.in );

        /* legge i due numeri interi a e b */
        System.out.println("Scrivi due numeri interi");
        /* legge due numeri interi a e b */
        a = in.nextInt();
        b = in.nextInt();
        /* calcola la somma di a e b e la visualizza */
        somma = a+b;
        System.out.print("La somma dei due numeri è ");
        System.out.println(somma);
    }
}

import fiji.io.*;
/* Applicazione che legge dalla tastiera due numeri interi
 * e ne calcola e visualizza la somma. */
public class SommaDueNumeri {
    public static void main(String[] args) {
        int a;        // il primo numero intero
        int b;        // il secondo numero intero
        int somma;    // la somma di a e b

        /* legge i due numeri interi a e b */
        System.out.println("Scrivi due numeri interi");
        /* legge due numeri interi a e b */
        a = Lettore.in.leggiInt();
        b = Lettore.in.leggiInt();
        /* calcola la somma di a e b e la visualizza */
        somma = a+b;
        System.out.print("La somma dei due numeri è ");
        System.out.println(somma);
    }
}
```

73

Oggetti e Java

Fondamenti di informatica: Oggetti e Java
Luca Cabibbo

Esercizio

Scrivere un'applicazione che legge dalla tastiera un numero razionale, ne calcola la radice quadrata e la visualizza sullo schermo

Scrivi un numero

1.21

La radice quadrata di 1.21 è 1.1

- per leggere dalla tastiera un numero razionale bisogna usare il metodo **double nextDouble()** di **Scanner**
 - oppure, il metodo **double leggiDouble()** di **Lettore.in**

74

Oggetti e Java

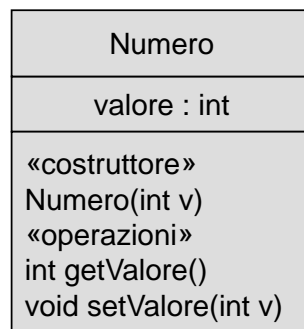
Fondamenti di informatica: Oggetti e Java
Luca Cabibbo

Una classe per istanziare oggetti

Si vuole definire la classe **Numero** per istanziare oggetti

- un oggetto **Numero** rappresenta un numero intero
- lo stato di un oggetto **Numero** consiste nel **valore** del numero intero
- per creare un oggetto **Numero** bisogna specificare il suo valore iniziale
- un oggetto **Numero** deve saper eseguire le seguenti operazioni
 - **int getValore()** – restituisce il valore del numero
 - **void setValore(int v)** – cambia il valore del numero in **v**

La classe Numero



Uso della classe Numero

```
/* Applicazione di prova per la classe Numero. */
public class DemoNumero {
    public static void main(String[] args) {
        Numero n1, n2;          // due numeri
        int v1, v2;            // valore di n1 e n2

        /* crea i due numero */
        n1 = new Numero(10);
        n2 = new Numero(20);

        /* calcola e visualizza il valore di n1 */
        v1 = n1.getValore();
        System.out.println(v1);          // 10

        /* calcola e visualizza il valore di n2 */
        v2 = n2.getValore();
        System.out.println(v2);          // 20

        /* cambia il valore di n1 e poi lo visualizza */
        n1.setValore(50);
        v1 = n1.getValore();
        System.out.println(v1);          // 50
    }
}
```

Classi per istanziare oggetti

Come si definisce una classe per istanziare oggetti?

- progetto (descrizione) di un oggetto
- dichiarazione di variabili d'istanza
 - per rappresentare lo stato (le proprietà)
- definizione di metodi d'istanza
 - per implementare le operazioni degli oggetti
- definizione di costruttori
 - per implementare le modalità di costruzione dalla classe

La classe Numero

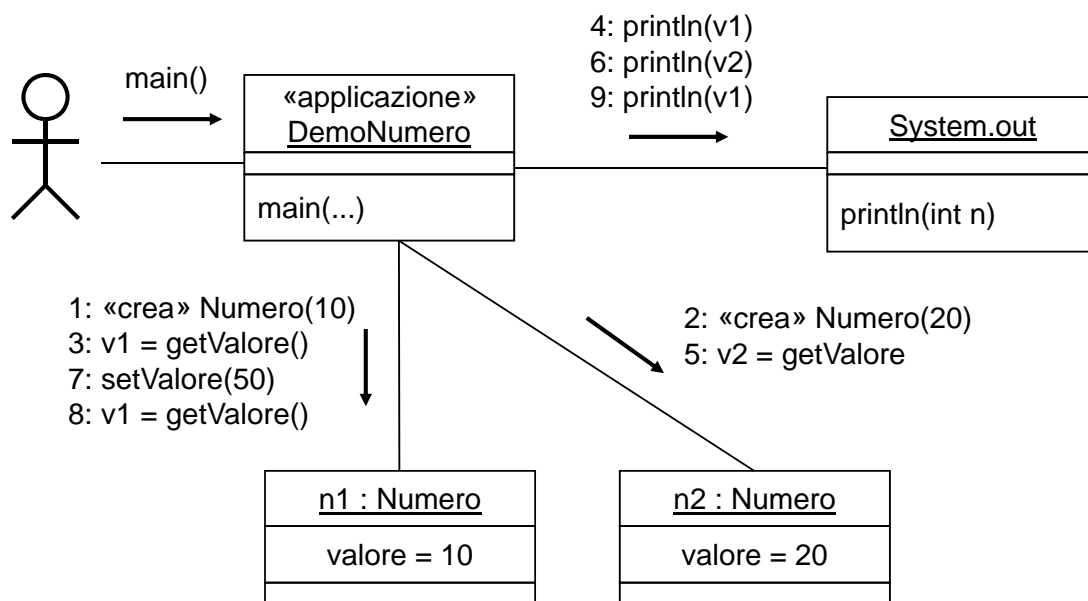
```
/* Un oggetto Numero rappresenta un numero intero,  
 * con un valore che può essere acceduto e modificato. */  
public class Numero {  
    /* valore del numero */  
    private int valore;  
  
    /* Crea un nuovo Numero che ha valore iniziale v. */  
    public Numero(int v) {  
        this.valore = v;  
    }  
  
    /* Calcola il valore di questo numero. */  
    public int getValore() {  
        return this.valore;  
    }  
    /* Cambia in v il valore di questo numero. */  
    public void setValore(int v) {  
        this.valore = v;  
    }  
}
```

79

Oggetti e Java

Fondamenti di informatica: Oggetti e Java
Luca Cabibbo

Diagramma di collaborazione per DemoNumero



80

Oggetti e Java

Fondamenti di informatica: Oggetti e Java
Luca Cabibbo