Corso di Laurea Ingegneria Informatica

Fondamenti di Informatica 1

Dispensa E01

Esempi di programmi

A. Miola

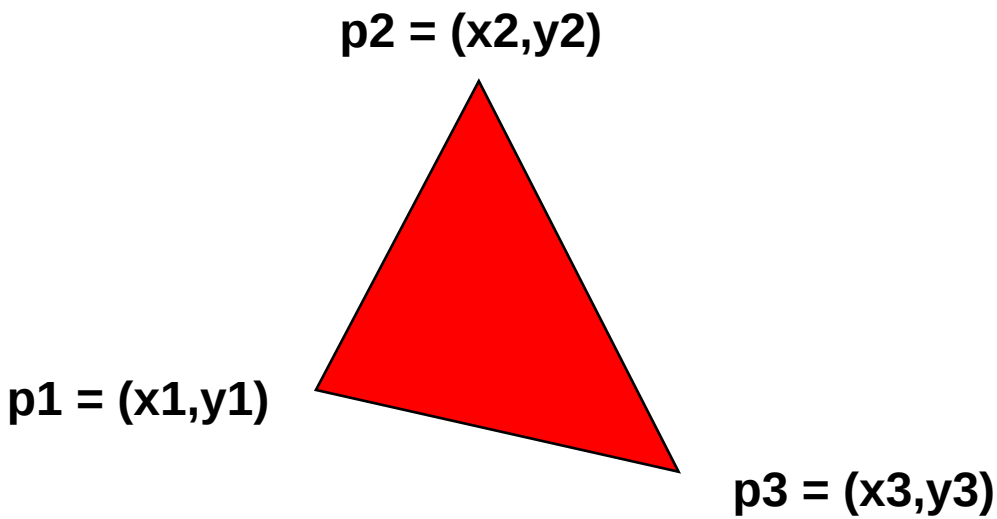
Settembre 2007

Contenuti

- Vediamo in questa lezione alcuni primi semplici esempi di applicazioni
- In particolare consideriamo ora due esempi presentati nel libro di testo, nei paragrafi 3.2.3 e 3.2.4, che avevamo omesso in precedenza
  - Calcolo del perimetro di un triangolo
  - Lettura e somma di due numeri interi

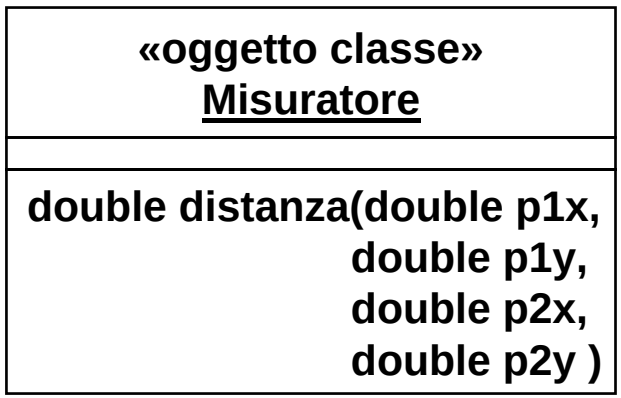
Perimetro di un triangolo

- Si vuole scrivere una applicazione che calcola e visualizza il perimetro di un triangolo, sul piano cartesiano, di cui sono noti i vertici

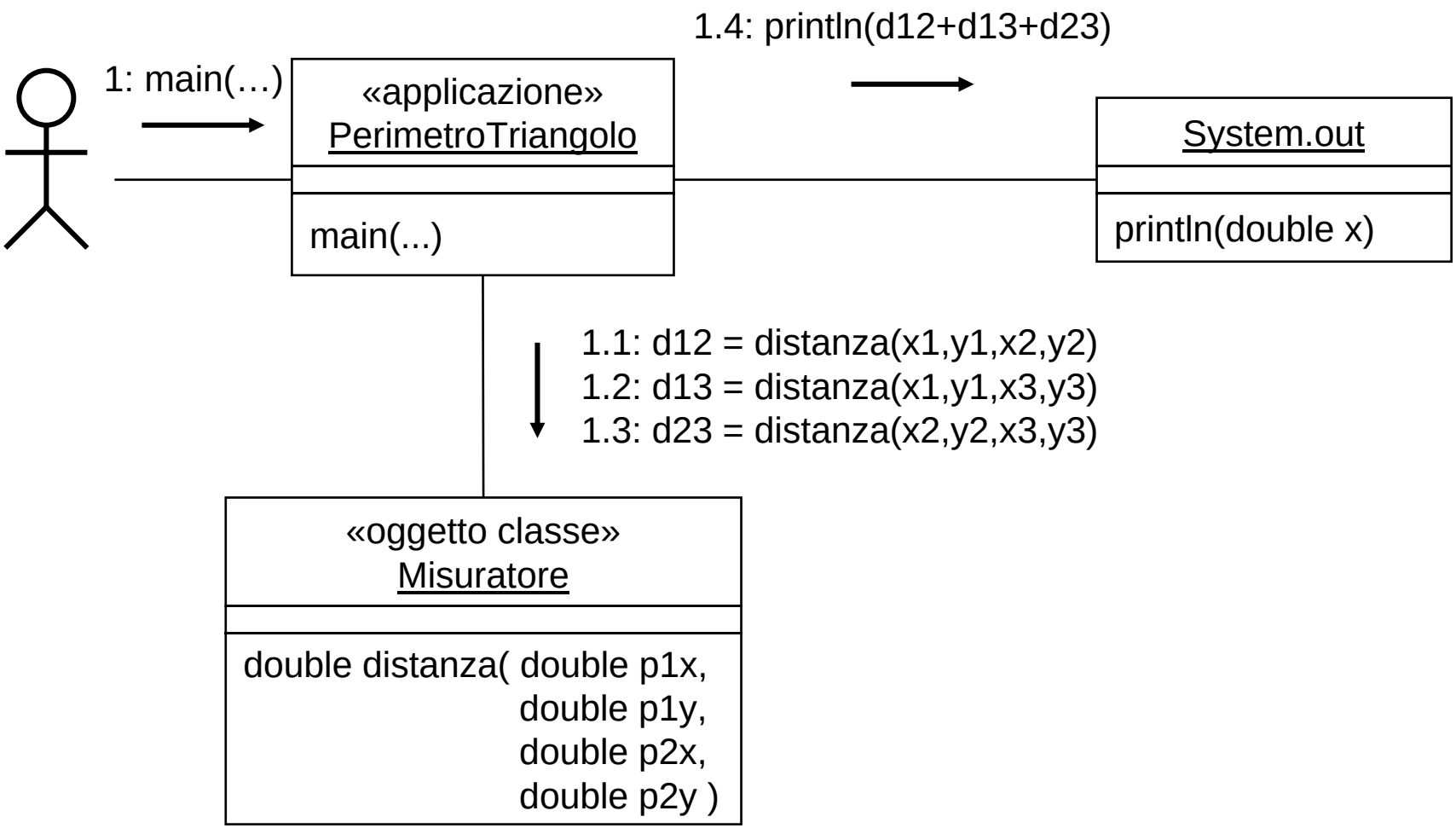


L’oggetto Misuratore

- Nella definizione dell’applicazione richiesta faremo uso di un oggetto classe chiamato **Misuratore** che, in base al suo comportamento e quindi alla sua interfaccia a noi nota, sa calcolare la distanza tra due punti, date le coordinate dei due punti



Collaborazione che si vuole realizzare



L'applicazione PerimetroTriangolo

```
/* Applicazione che calcola e visualizza
 * il perimetro di un triangolo. */
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;
```

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

... segue ...

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, ma 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**
- **Si deve allora definire l’oggetto Misuratore**
  - **bisogna definire una classe per l’oggetto Misuratore**
  - **questo oggetto, come previsto, deve saper eseguire l’operazione distanza**
- **La definizione dell’applicazione PerimetroTriangolo richiede quindi la definizione di due classi**

La classe Misuratore

```
/* Oggetto che sa calcolare la distanza tra due punti. */
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;
    }
}
```

Parametri formali e parametri attuali

- **Definizione del metodo distanza**

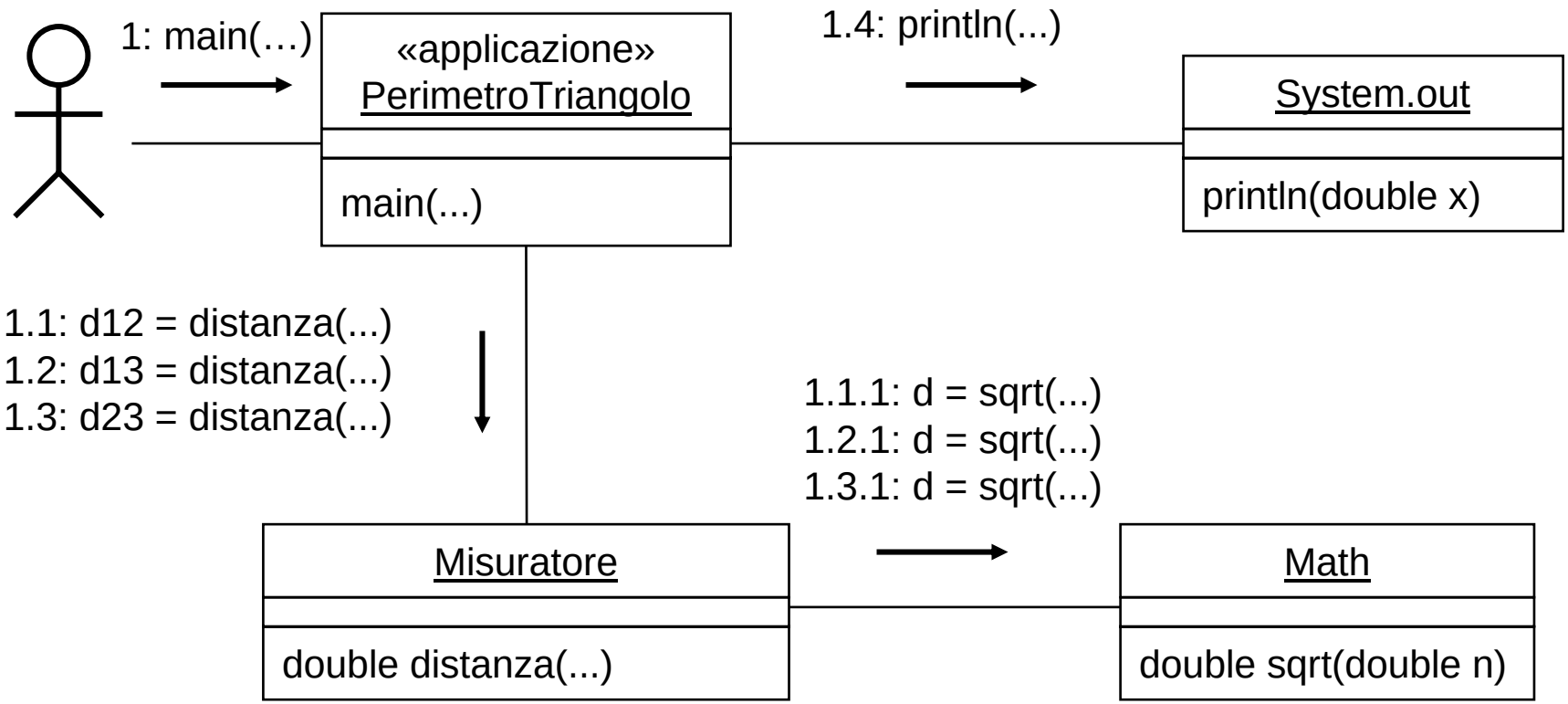
  ```
  public static double distanza (double p1x, double p1y,
                                 double p2x, double p2y) { . . . }
  ```

- **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**
  - le variabili dichiarate come parametri del metodo sono i parametri formali del metodo
  - i valori specificati in una richiesta di esecuzione del metodo sono i parametri attuali per l'esecuzione del metodo
  - legame dei parametri – in numero, ordine e tipo

Diagramma di collaborazione per PerimetroTriangolo



Lettura e somma di due numeri interi

- **Si vuole scrivere una 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 è troppo difficile da usare
- In questo corso viene usato l'oggetto Lettore.in, del package fiji.io disponibile sul sito web

Lettore.in
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. */

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");
. . .
```

. . . L’applicazione SommaDueNumeri

```
. . .
        /* 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 una 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() dell'oggetto Lettore.in**

Esercizio

- Scrivere una applicazione Java che calcola il perimetro di un triangolo leggendo dalla tastiera tre coppie di numeri razionali, che rappresentano le coordinate dei suoi vertici, e visualizza il risultato sullo schermo

Istruzioni base per semplici applicazioni

- **Per il momento ci stiamo limitando a vedere (leggere e scrivere) semplici applicazioni basate sull'uso delle seguenti tipologie di istruzioni**
  - **istruzioni semplici (*ne abbiamo già visto l'uso*)**
    - **invio di un messaggio a un oggetto**
    - **calcolo di una espressione e assegnazione**
  - **istruzioni di controllo (*iniziamo ad usarle adesso*)**
    - **sequenza — una sequenza finita di istruzioni**
    - **istruzione condizionale — esegue una istruzione condizionatamente al verificarsi di una condizione**
    - **istruzione ripetitiva — esegue una istruzione ripetutamente al verificarsi di una condizione**

Sequenza

- **Una sequenza (o istruzione composta o blocco) è un gruppo di istruzioni che devono essere eseguite in sequenza – una alla volta, una dopo l'altra**
- **Sintassi**

  { ***istruzione-1***

  istruzione-2

  ...

  istruzione-n **}**

- **Semantica (significato)**
  - **esegui *istruzione-1*, poi *istruzione-2*, ..., poi *istruzione-n***

Istruzione condizionale

- **L'istruzione condizionale (istruzione if-else) consente di eseguire una istruzione tra una coppia di istruzioni condizionatamente al verificarsi (o meno) di una condizione**
- **Sintassi**

```
if (condizione)
        istruzione-1
    else
        istruzione-2
```

- **Semantica (significato)**
  - **valuta la condizione *condizione***
  - **se la condizione *condizione* si è verificata allora esegui l'istruzione *istruzione-1*, altrimenti esegui l'istruzione *istruzione-2***

Istruzione ripetitiva

- **L'istruzione ripetitiva (istruzione while) consente di eseguire ripetutamente una istruzione fintanto che una condizione risulta verificata**
- **Sintassi**

  while (*condizione*)

  istruzione
- **Semantica (significato)**
  - **esegui ripetutamente e in sequenza questi due passi**
    - **valuta la condizione *condizione***
    - **se la condizione *condizione* si è verificata allora esegui l'istruzione *istruzione*, altrimenti termina**

Lettura e prodotto di due numeri interi

- **Si vuole scrivere una applicazione Java che legge dalla tastiera due numeri interi, ne calcola il prodotto e lo visualizza sullo schermo**
- **Si richiede di risolvere il problema in due modi**
  - **utilizzando l'operazione di prodotto tra interi disponibile attraverso l'operatore ***
    - **assumiamo questo come un esercizio veramente semplice**
  - **utilizzando l'algoritmo visto in precedenza per ottenere il valore del prodotto attraverso una iterazione di operazioni di somma**
    - **si tratta di scrivere un'applicazione Java a partire dal seguente algoritmo rappresentato in pseudo-codice**

L’algoritmo per il Prodotto

```
Prodotto
        integer x, y, m, n, z;
        input (x, y);
        if (x > y) {
              m = x; n = y;
        } else {
              m = y; n = x;
        };
        z = 0;
        while (n > 0) {
              z = z + m;
              n = n - 1;
        }
        output (z);
```

Esercizio

- **Scrivere una applicazione Java che legge dalla tastiera due numeri interi, ne calcola il quoziente e lo visualizza sullo schermo**
- **Si richiede di risolvere il problema in due modi**
  - **utilizzando l'operazione di quoziente tra interi disponibile attraverso l'operatore /**
    - **assumiamo questo come un esercizio veramente semplice**
  - **definendo prima un'algoritmo per ottenere il valore del quoziente attraverso una iterazione di operazioni di sottrazione**

Riferimenti al libro di testo

- **Per lo studio di questi argomenti si fa riferimento al libro di testo, e in particolare al capitolo**
  - **3 su Oggetti e Java**
- **Con particolare riferimento ai seguenti paragrafi**
  - **3.2.3 – Perimetro di un triangolo**
  - **3.2.4 – Lettura e somma di due numeri interi**