Scopri l'OOP in Java

Terzo incontro, 22/01/2025

Prof.sse A. Schiavon, L. Dalla Montà, gennaio 2025

Scopri l'OOP in Java

Lezione 3, 22/01/2025

- Classi dell'API di Java, uso delle classi più comuni
  - Random
  - Rectangle

- Dall'ADT alla classe Java, attributi, metodi
  - Punto
  - Palloncino

- Documentare il codice Java con javadoc
  - https://www.spacecoding.it/javadoc-documentazione-del-codice-java/

Classe Random - uno sguardo alla documentazione API

java.util

Class Random

java.lang.Object
    java.util.Random

All Implemented Interfaces:

Serializable

Direct Known Subclasses:

SecureRandom, ThreadLocalRandom

Constructor Summary

Constructors

Constructor and Description
Random() Creates a new random number generator.
Random(long seed) Creates a new random number generator using a single long seed.

boolean	**nextBoolean**() Returns the next pseudorandom, uniformly distributed boolean value from this random number generator's sequence.
void	**nextBytes**(byte[] bytes) Generates random bytes and places them into a user-supplied byte array.
double	**nextDouble**() Returns the next pseudorandom, uniformly distributed double value between 0.0 and 1.0 from this random number generator's sequence.
float	**nextFloat**() Returns the next pseudorandom, uniformly distributed float value between 0.0 and 1.0 from this random number generator's sequence.
double	**nextGaussian**() Returns the next pseudorandom, Gaussian ("normally") distributed double value with mean 0.0 and standard deviation 1.0 from this random number generator's sequence.
int	**nextInt**() Returns the next pseudorandom, uniformly distributed int value from this random number generator's sequence.
int	**nextInt**(int bound) Returns a pseudorandom, uniformly distributed int value between 0 (inclusive) and the specified value (exclusive), drawn from this random number generator's sequence.
long	**nextLong**() Returns the next pseudorandom, uniformly distributed long value from this random number generator's sequence.

Classe Random

```java
import java.util.Random;
public class RandomTester{
    public static void main(String[] args){
        Random r1 = new Random();
        System.out.println(r1.hashCode());
        System.out.println(r1.toString());

        // estrazione di un valore intero (reale, booleano, ...) casuale
        int numeroCasuale=r1.nextInt();
        System.out.println(numeroCasuale);

        // estrazione di un secondo nuemro casuale
        System.out.println(r1.nextInt());
    }
}
```

Considera l'esempio in figura: a tuo parere, quali sono i valori possibili per il numero casuale estratto? Perché?

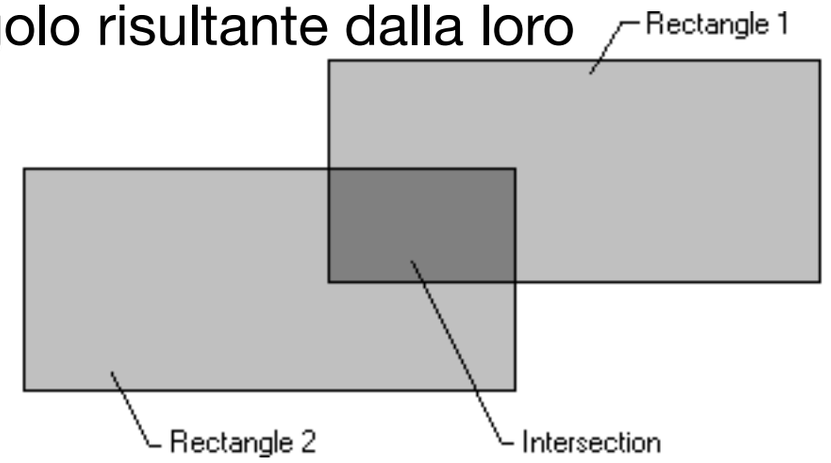
Attività: cercare nella documentazione API i metodi della classe Random per:

- Estrarre un numero casuale intero compreso tra -5 e +5
- Estrarre 5 valori booleani
- Estrarre 10 numeri reali compresi tra 0.0 e 1.0
- Simulare il gioco del lancio di un dado:
  - la vittoria c'è se al lancio si ottiene il numero 6
  - se dopo 5 lanci non è mai uscito il 6, la partita è persa.

Scopri le classi Rectangle e Point

Consultare la documentazione API (package java.awt) e, scegliendo i metodi corretti, svolgere le seguenti attività.

- Realizzare l'UML della classe (sintesi)
- Realizzare una classe tester, per:
  - ✓ Istanziare un oggetto Rectangle
  - ✓ Istanziare tre oggetti Rectangle, i cui vertici in alto a sinistra corrispondano a:
    - le coordinate (0,0), con base e altezza 0
    - un oggetto Point(10,20), con base e altezza 0
    - il punto di coordinate x=5 y=5, con base 15 e altezza 5
  - ✓ Scegliere uno degli oggetti Rectangle con dimensione 0 e aumentarne base e altezza, con valori a scelta
  - ✓ Scegliere due degli oggetti Rectangle e restituire il rettangolo risultante dalla loro intersezione



  - ✓ Dopo ciascuna azione eseguita, per ogni oggetto sopracitato, visualizzarne lo stato.

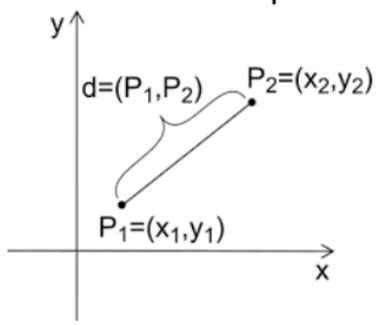
Progettare e implementare classi ADT

Esempio
l'ADT Punto

Realizzare l'ADT che descrive un Punto.
Le **caratteristiche** (=attributi) di un punto P(x,y) sono le sue coordinate (x, y), cioè i valori sull'asse delle ascisse (x) e delle ordinate (y) di un piano cartesiano.
L'ADT Punto deve comprendere le seguenti **funzionalità** (=metodi):

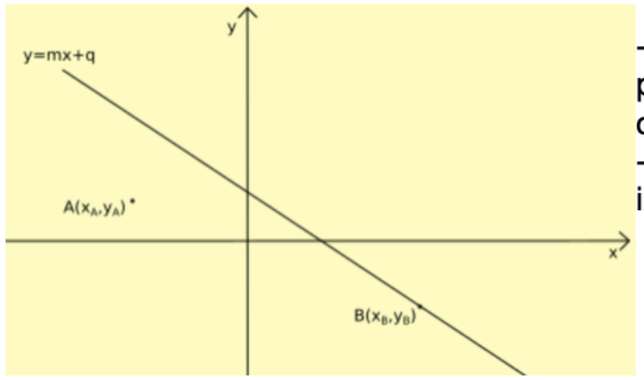
(a) confronto del punto con un secondo punto, per determinare se sono uguali (due punti P1 e P2 sono uguali se x1=x2 e y1=y2)

(b) calcolo della distanza ***d*** del punto rispetto ad un altro punto:



$$d(P_1, P_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

(c) data l'equazione di una retta y=mx+q, cioè fissati il coefficiente m e il termine noto q, verificare se il punto appartiene a quella retta. Ecco alcuni suggerimenti:



- Il punto A non appartiene alla retta, e lo si può verificare calcolando mx_A+q e constatando che $y_A \neq mx_A+q$
- Invece il punto B appartiene alla retta, infatti $y_B=mx_B+q$

Realizzare:

1) il diagramma UML dell'ADT Punto, con gli attributi, il metodo costruttore, i metodi **(a) *distanza***, **(b) *uguale*** e **(c) *appartieneA*** descritti sopra e il metodo toString;
2) la classe Java Punto, derivata dall'UML, implementando attributi e metodi;
3) una classe di collaudo con le istruzioni per:
   a. istanziare due oggetti punto P1(2,1) e P2(3,2)
   b. calcolare e visualizzare la distanza tra il punto P1 e il punto P2
   c. verificare se un punto (P1 o P2) appartiene alla retta y=2x-4, visualizzando un opportuno messaggio
   d. verificare se il punto P2 sia uguale al punto P1, visualizzando un opportuno messaggio.

Progettare e implementare classi ADT

Scrivere il diagramma di classe ed implementare le classi Java per il seguente caso

Palloncino

Si vuole realizzare un programma che controlla un pallone gonfiabile e il processo di riempimento, nel senso che un pallone possiede una capienza massima, superata la quale, il pallone scoppia; inoltre si vogliono calcolare anche la misura del raggio e la misura dell'area della superficie del pallone, una volta gonfiato.
Alla fine si desidera sapere quanti palloni sono stati gonfiati.

Realizzare la classe ***Palloncino*** con attributi
- la ***capacità massima*** del pallone
- il ***volume*** del pallone, una volta gonfiato
- la numerosità dei palloni.

Completare la classe con i commenti javadoc.

Realizzare la classe di collaudo ***PalloncinoTester***, che istanza alcuni palloncini e ne verifica i metodi.