

# Classi ed Oggetti

Fondamenti di Informatica A-K

# Esercitazione 5

Introduzione al calcolatore e Java

Linguaggio Java, basi e controllo del flusso

I metodi: concetti di base

Stringhe ed array

Classi e oggetti, costruttori, metodi statici, visibilità

Eclipse, ereditarietà e polimorfismo

# Classi e oggetti

Java è un linguaggio totalmente orientato agli oggetti (Object Oriented).

A prescindere dall'oggetto che si vuole modellare una classe sarà sempre costituita da:

- Dati → Attributi
- Metodi → Azioni

# Esempio di Classe

```
class Automobile
```

Come modellare il tipo di oggetto automobile attraverso una serie di attributi e metodi?

## Attributi

Targa

Indicatore  
carburante

Velocità

## Metodi

Modifica targa

Accellera

Decelera

# Implementazione in Java: classe

```
Public class Automobile {  
  
    private double carburante = 0;  
    private double velocità = 0;  
    private String targa = "";  
  
    public void modificaTarga (String nuova_targa) {  
        targa = nuova_targa;  
    }  
  
    public void accelera (double accelerazione) {  
        velocità += accelerazione;  
    }  
  
    public void decelera (double decelerazione) {  
        velocità -= decelerazione;  
    }  
}
```

# Implementazione in Java: attributi

```
Public class Automobile {
```

```
    private double carburante = 0;  
    private double velocità = 0;  
    private String targa = "";
```

Attributi

```
    public void modificaTarga (String nuova_targa) {  
        targa = nuova_targa;  
    }
```

```
    public void accelera (double accelerazione) {  
        velocità += accelerazione;  
    }
```

```
    public void decelera (double decelerazione) {  
        velocità -= decelerazione;  
    }
```

```
}
```

# Implementazione in Java: metodi

```
Public class Automobile {
```

```
    private double carburante = 0;  
    private double velocità = 0;  
    private String targa = "";
```

Metodi

```
    public void modificaTarga (String nuova_targa) {  
        targa = nuova_targa;  
    }
```

```
    public void accelera (double accelerazione) {  
        velocità += accelerazione;  
    }
```

```
    public void decelera (double decelerazione) {  
        velocità -= decelerazione;  
    }
```

```
}
```

# I costruttori: metodi speciali

Si chiama costruttore uno speciale metodo che non ha tipo di ritorno ed il cui nome coincide con quello della classe

Un costruttore viene automaticamente invocato quando una classe viene istanziata

```
Automobile Oggetto1 = new Automobile("Ak 147 ", 50 , 15 );
```



# Implementazione in Java

```
Public class Automobile {
```

```
    private double carburante = 0;  
    private double velocità = 0;  
    private String targa = "";
```

Costruttore

```
    public Automobile (String targa, double velocità, double  
                      carburante) {  
        this.targa = targa;  
        this.velocità = velocità;  
        this.carburante = carburante;  
    }
```

```
    public void modificaTarga (String nuova_targa) {  
        targa = nuova_targa;  
    }
```

```
    ...altri metodi...
```

```
}
```

# Utilizzare i metodi

```
Public class Main{  
    public static void Main (String[] args){  
        Automobile auto1 = new Automobile("Ak 147 ", 50 , 15 );  
        Automobile auto2 = new Automobile("RM 164 ", 120 , 19 );  
        auto1.accellera(5);  
        auto2.decellera(10);  
    }  
}
```

Per accedere a metodi ed attributi pubblici di un oggetto basta utilizzare la notazione puntata.

# Esercizio «MiaVariabile»

- Creare la classe «MiaVariabile» dotata di
  1. Un campo intero chiamato "valore"
  2. Un metodo **getValore()** che restituisca il valore il "valore"
  3. Un metodo **setValore(int)** che imposti il valore di "valore"
  4. Un metodo **resetValore()** che azzeri il valore di "valore"
- Creare la classe «Main» che
  1. Definisca il metodo main
  2. Crei un'istanza della classe MiaVariable
  3. Chiami il metodo **resetValore()** sull'istanza
  4. Stampi a video il valore attuale di "valore"
  5. Chiami il metodo **setValore(8)** sull'istanza
  6. Stampi a video il valore utilizzando il metodo **getValore()**

# Variabili e Metodi **static**

## Istanziare Oggetto

- Per accedere agli attributi o ai metodi di una classe bisogna prima istanziare l'oggetto corrispondente.

## Metodi Statici

- Se un metodo non ha alcuna relazione con l'oggetto, qualunque sia il suo tipo, può essere definito staticamente.

## Classe VS. Oggetto

- I dati appartenenti ad una classe statica si riferiscono alla classe. I dati comuni invece, non statici, appartengono all'oggetto.

# Esempio Variabili e Metodi **static**

La classe `MiaVariabile2` dichiara un unico campo privato e quindi non accessibile dall'esterno.

```
public class MiaVariabile2 {  
    private static int valore;  
    public static int getValore () {  
        return valore;  
    }  
    public static void setValore (int nuovoValore) {  
        valore = nuovoValore;  
    }  
}
```

Il campo `valore` è associato alla classe ed è unico all'interno della JVM.

I metodi devono essere definiti statici per poter essere associati alla classe

# Esercizio Variabili e Metodi **static**

- Definire una classe Main2 che utilizzi la classe MiaVariabile2 appena definita senza istanziarla.
- Ad esempio si può provare a:
  1. Settare il valore della variabile a 3
  2. Sommarla ad un intero inizializzato a 4
  3. Definire un intero "somma" che contenga la somma.
  4. Stampi la somma

# Creazione di un Array di oggetti

- E' possibile definire array di qualsiasi tipo di oggetto (purché sia lo stesso oggetto).
  - Possiamo creare un array di interi, di stringhe, di classi Persona, di classi Contatore.
- Il codice in basso non funziona, lancia un **NullPointerException** perché sta tentando di accedere ad un oggetto non ancora inizializzato.

```
public class TestArrayPersona {  
    public static void main (String args[]) {  
        Persona[] persona = new Persona[5];  
        persona[0].nome;  
    }  
}
```

# Istanziare un Array di oggetti

Per farlo funzionare bisogna istanziare ogni singolo oggetto con new!!

```
public class TestArrayPersona {  
    public static void main (String args[]) {  
        Persona[] persona = new Persona[5];  
        for(int i=0; i<contatori.length; i++){  
            persona[i] = new Persona();  
        }  
        persona[0].nome;  
    }  
}
```



# Esercizio Biblioteca (homework)

- Obiettivo: si vuole scrivere un programma per gestire libri e prestiti di una biblioteca.
- La classe deve contenere almeno i seguenti attributi:
  1. Un array di titoli di libri
  2. Un array booleano che tenga conto per ciascun libro se è prestato o meno
  3. Un intero che tenga conto del numero totale di libri
  4. Un intero che tenga conto del numero totale di prestiti
- La classe deve contenere almeno i seguenti metodi:
  1. Un costruttore che inizializza gli array con una dimensione massima a scelta.
  2. Un metodo per l'aggiunta di un libro
  3. Un metodo per il prestito di un libro
  4. Un metodo per la restituzione di un libro
  5. Un metodo di ricerca di un titolo all'interno dell'array dei titoli che ritorna la posizione del libro (-1 se non c'è)