Introduzione alla programmazione in Java

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

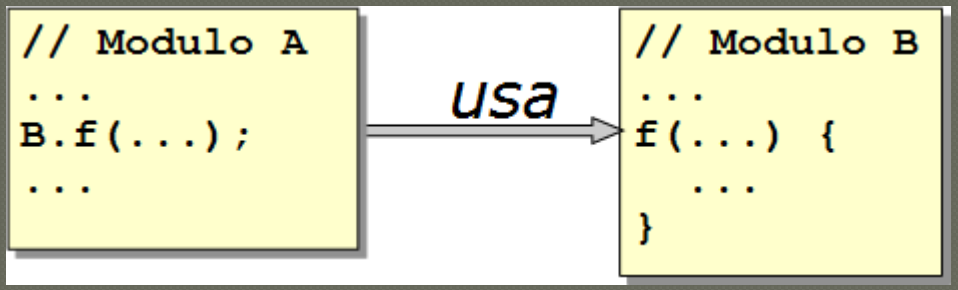
Il concetto di modulo

- con questo nome si indica in genere un "pezzo" di programma, dotato di una propria identità: un modulo, ad esempio, potrebbe essere l'insieme di alcune funzioni C, o Pascal, che risiedono in un unico file sorgente; oppure il modulo può essere già compilato, ed essere disponibile in una libreria per venire incluso (riusato) nel codice eseguibile.
- Programmare significa quindi "aggregare" moduli diversi, riusando le funzionalità che ci mettono a disposizione; scriveremo direttamente del codice solo per ***specializzare un modulo adattandolo alle nostre particolari*** necessità, per ***integrare (cioè connettere tra loro) i vari moduli, oppure*** per scrivere un modulo da zero se non ne avremo trovato uno già pronto che faccia al caso nostro.

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Struttura modulare dei programmi

In generale i nostri programmi saranno costituiti da una rete di moduli, connessi da relazioni d'uso: se il codice del modulo *A ad un certo punto* chiama una funzione *f contenuta nel modulo B, diciamo che A usa B. La* rete dei moduli che rappresenta l'architettura del programma avrà spesso (ma non sempre) una struttura gerarchica.

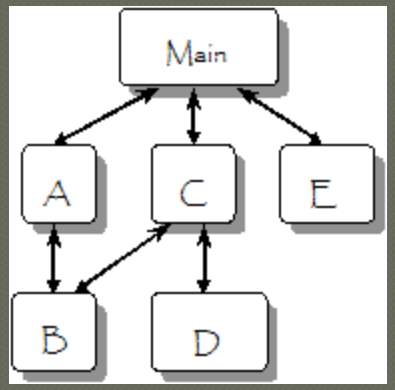


Nella programmazione OO il concetto di *modulo si sovrappone,* parzialmente, a quello di *oggetto.*

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Gli oggetti

Nel paradigma OO i programmi sono costituiti da una rete di oggetti connessi da relazioni d'uso.



Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Definizione 1 di oggetto

Un oggetto è un'area di memoria inizializzata

Java garantisce che le aree di memoria associate ad un oggetto vengano inizializzate subito dopo la creazione dell'oggetto, attraverso l'attivazione automatica di una particolare componente di codice detta **costruttore**.

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Definizione 2 di oggetto

Un oggetto è un'aggregazione di variabili e metodi

Questa definizione ci porta a comprendere che "dentro" un oggetto ci sono cose eterogenee:

- **dati (variabili) ;**
- **comportamenti (metodi, cioè funzioni).**

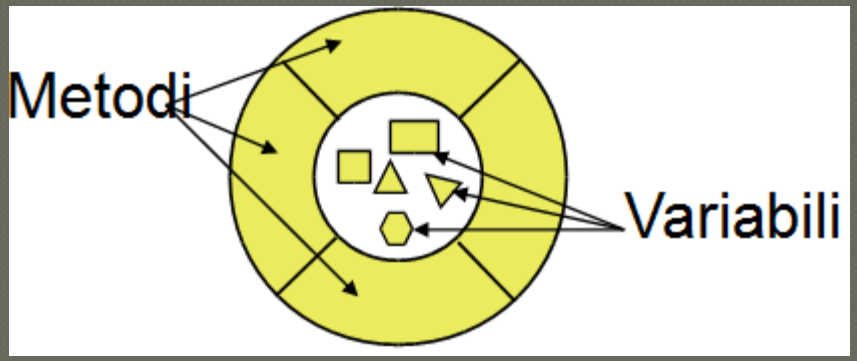
Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Definizione 2 di oggetto

Un oggetto è un'aggregazione di variabili e metodi

I metodi ***incapsulano l'oggetto: le variabili non sono*** direttamente accessibili dall'esterno, ma possono essere manipolate solo usando opportuni metodi.

Questo meccanismo ci consente di ***nascondere i dati ed esporre solo (alcuni)*** metodi. I servizi offerti dall'oggetto sono accessibili solo attraverso ***un'interfaccia funzionale.***



Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Definizione 3 di oggetto

Un oggetto è l'istanza di una classe

Questa invece è una definizione più difficile da comprendere, anche perché non abbiamo ancora visto in dettaglio che cos'è una "classe" e che cosa vuol dire "istanziare". La struttura e il comportamento di un oggetto sono dichiarati in una **classe, che è un'estensione della struct del** linguaggio C o del **record del Pascal.**

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Gli oggetti

Gli **oggetti** rappresentano le entità del problema o della realtà che si vuole automatizzare con l'informatica. Un **oggetto** è in grado di memorizzare le informazioni che riguardano il suo stato; è anche possibile associare a un **oggetto** un insieme di operazioni che esso può compiere.

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Le classi

La **classe** è la descrizione astratta degli oggetti attraverso gli attributi e i metodi.

Una classe "rappresenta" uno o più oggetti, tutti fatti a sua immagine: la classe raccoglie tutto ciò che tali oggetti hanno in comune.
Una classe è un ***tipo, in quanto fornisce una descrizione sia strutturale (le*** variabili) che *funzionale (i metodi, cioè tutte le operazioni che si possono* effettuare su un dato). Le ***variabili (slot, data member, campi) rappresentano lo stato dell'oggetto e sono spesso dichiarate private*** per "nasconderle" all'interno dell'oggetto: l'implementazione del servizio è un fatto privato della classe, per impedire che modifiche ed errori vengano propagati al codice esterno. Ogni oggetto istanza di una classe ha un suo set di variabili (fanno eccezione le variabili dichiarate **static che** sono condivise da tutte le istanze di una classe).

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

I metodi

I metodi (*member function) sono in genere dichiarati* ***public: essi*** costituiscono ***l'interfaccia esposta della classe, che include tutti e solo i*** metodi che possono essere usati per manipolare le istanze di quella classe.
Per usare i servizi offerti dalla classe l'unica cosa che il codice *cliente deve* conoscere è l'*interfaccia: di ogni metodo* ***public deve essere nota la*** *signature, cioè il nome del metodo, il numero e tipo dei parametri di* chiamata, il tipo del valore di ritorno.
Un metodo può essere applicato solo su istanze della classe in cui è definito (fanno eccezione i metodi **static). Tutti gli oggetti della classe** condividono i suoi metodi.

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Istanziare una classe

Istanziare un oggetto di una determinata classe significa:

- dichiarare una variabile di quel tipo;
- allocare dinamicamente spazio per un oggetto di quel tipo.

Esempio:

Integer x = new Integer(10);

Nell'esempio dichiariamo una variabile **x di tipo Integer, e le** assegniamo il riferimento ad una nuova istanza della classe **Integer, allocata dinamicamente e inizializzata col valore 10.**

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

I costruttori

Una classe può definire uno o più *costruttori.*

- Un costruttore è una componente di codice che viene chiamata automaticamente quando la classe viene istanziata, e provvede all'inizializzazione dell'oggetto.
- Un costruttore è simile a un metodo, ma ne differisce per due aspetti:
  - Un costruttore non viene mai chiamato esplicitamente, ma viene attivato dal sistema in fase di inizializzazione dell'oggetto.
  - Un costruttore non ritorna alcun valore.

Nell'esempio:

Integer x = new Integer(10);

L'operatore **new fa "nascere" un nuovo oggetto istanza della classe Integer, e chiama il costruttore passandogli il valore 10. Il costruttore** provvede ad inizializzare l'oggetto con tale valore.

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Attributi e metodi

Gli **attributi** rappresentano gli elementi che caratterizzano l'oggetto, utili per descrivere le sue proprietà e definirne il suo stato.
I **metodi** rappresentano le funzionalità (azioni che si possono compiere) che l'oggetto mette a disposizione.

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Glossario

Classe	Una classe è la definizione strutturale e funzionale di un tipo di oggetto. Una classe descrive un insieme di oggetti che condividono struttura e comportamento.
Oggetto	Un oggetto è un'area di memoria inizializzata, che aggrega variabili e metodi definiti nella classe* di cui l'oggetto è istanza. Un oggetto ha un'*interfaccia*.
Istanziare	Istanziare un oggetto di una determinata classe significa allocare dinamicamente spazio di memoria per le variabili definite in tale classe*, inizializzarne i valori e ottenere il riferimento all'oggetto come un valore il cui tipo è compatibile con la classe in questione.
Metodo	Un metodo è una funzione definita all'interno di una classe. Un metodo solo sugli oggetti istanze di tale classe*.
Interfaccia	Un'interfaccia è l'insieme delle *dichiarazioni* (o *signature*) dei metodi pubblici di una classe. La *signature* di un metodo comprende il nome del metodo, il tipo del valore ritornato, il numero e tipo dei parametri di chiamata, ma *non* il corpo del metodo (*implementazione*).
Costruttore	Un costruttore è una componente di codice che inizializza le variabili dell'oggetto.

N. B. Alcune di queste definizioni saranno meglio comprese, quando sarà introdotto il concetto di ereditarietà

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Come si dichiara una classe

esempio:

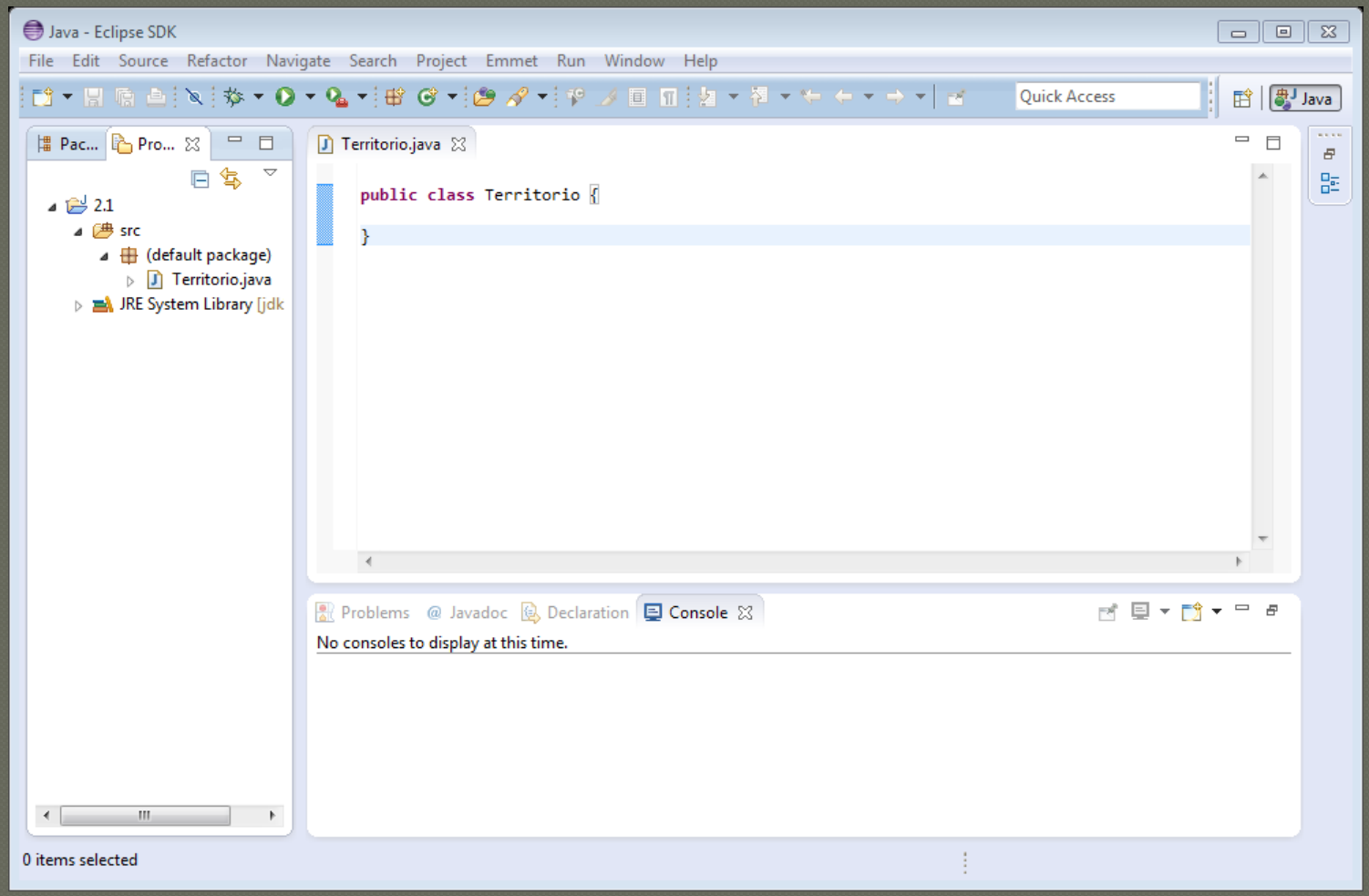
```
public class Territorio {
}
```

La definizione qui sopra è vuota, ma sintatticamente corretta. Il file sorgente che la contiene deve avere lo stesso nome della classe, ed estensione ".java"

Territorio.java

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

La classe Territorio in eclipse



Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Definizione delle variabili nella classe Territorio

```
public class Territorio
{
double superficie;
int abitanti;
}
```

Ogni istanza della classe **Territorio avrà due campi: la** superficie (in kmq) e il numero di abitanti. Nota che:.

- La variabile **superficie è definita di tipo double, e potrà** contenere valori numerici in virgola mobile con doppia precisione. Il tipo **double è *primitivo (built-in) in Java (come int, boolean,* char, float…) e *non è una classe.***
- La variabile **abitanti è definita di tipo int, e potrà contenere valori** numerici interi.

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Convenzioni per i nomi di classe, variabile, tipo e metodo

Nell'esempio appena visto le classi hanno nomi che iniziano con lettera maiuscola, mentre le variabili, i tipi *built-in, gli attributi che si riferiscono a metodi ed oggetti,* hanno nomi che iniziano con lettera minuscola:

```
public class Territorio
{
double superficie;
int abitanti;
}
```

Nel caso gli identificatori sono formati da più parole, queste vengono unite e ogni parola successiva alla prima avrà l'iniziale maiuscola, es. totFattura, contaSecondi, calcolaMedia.
Java è *case sensitive! nome, Nome, NOME, sono considerati diversi.*

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Definiamo un costruttore nella classe Territorio

```
public class Territorio
{
double superficie;
int abitanti;
Territorio(double s, int a)
        {
        superficie = s;
        abitanti = a;
        }
}
```

Il costruttore ha lo stesso nome della classe, e una lista di argomenti (parametri formali) che corrisponde ai valori (parametri attuali) che verranno specificati quando la classe verrà istanziata:
Territorio t = new Territorio(257.3, 12250);
Il costruttore inizializza il nuovo oggetto con i valori **257.3 per la superficie, e 12250** per gli abitanti.

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Definiamo un metodo nella classe Territorio

```
public class Territorio
{
double superficie;
int abitanti;
Territorio(double s, int a)
        {
        superficie = s;
        abitanti = a;
        }
double densita()
        {
        return abitanti / superficie;
        }
}
```

Per ogni istanza della classe **Territorio potrà essere chiamato il metodo densita(). Se t è il riferimento ad un oggetto di classe Territorio, possiamo** calcolare la densità abitativa di **t con l'espressione:**
t.densita()
L'operatore "." *applica il metodo* ***densita() all'oggetto t.***

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Il metodo main()

Affinché la classe **Territorio possa diventare un programma** eseguibile è necessario definire un metodo fatto così:

```
public static void main(String args[])
{
…
}
```

Il metodo **main() viene automaticamente eseguito all'attivazione del** programma, passandogli i parametri specificati a riga di comando.

L'interfaccia del metodo **main()** è **critica**: non specificarla o specificarla diversamente causerà un errore al momento della compilazione o dell'esecuzione.

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Istanziare una classe

Nel corpo del **main() istanziamo un oggetto della classe Territorio, e visualizziamo i valori dei campi superficie e abitanti, e il risultato della chiamata al metodo densita():**

```
public static void main(String args[])
{
Territorio t = new Territorio(257.3, 12250);
System.out.println(t.superficie);
System.out.println(t.abitanti);
System.out.println(t.densita());
}
```

- Il metodo **System.out.println() visualizza il valore del suo** argomento su *console.*

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

In generale

In ogni classe avremo:

- La dichiarazioni delle variabili;
- La definizione del costruttore;
- La definizione del metodo;
- Definizione del metodo main (critico);
- definizione delle istanza;
- Dati di input e output;

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

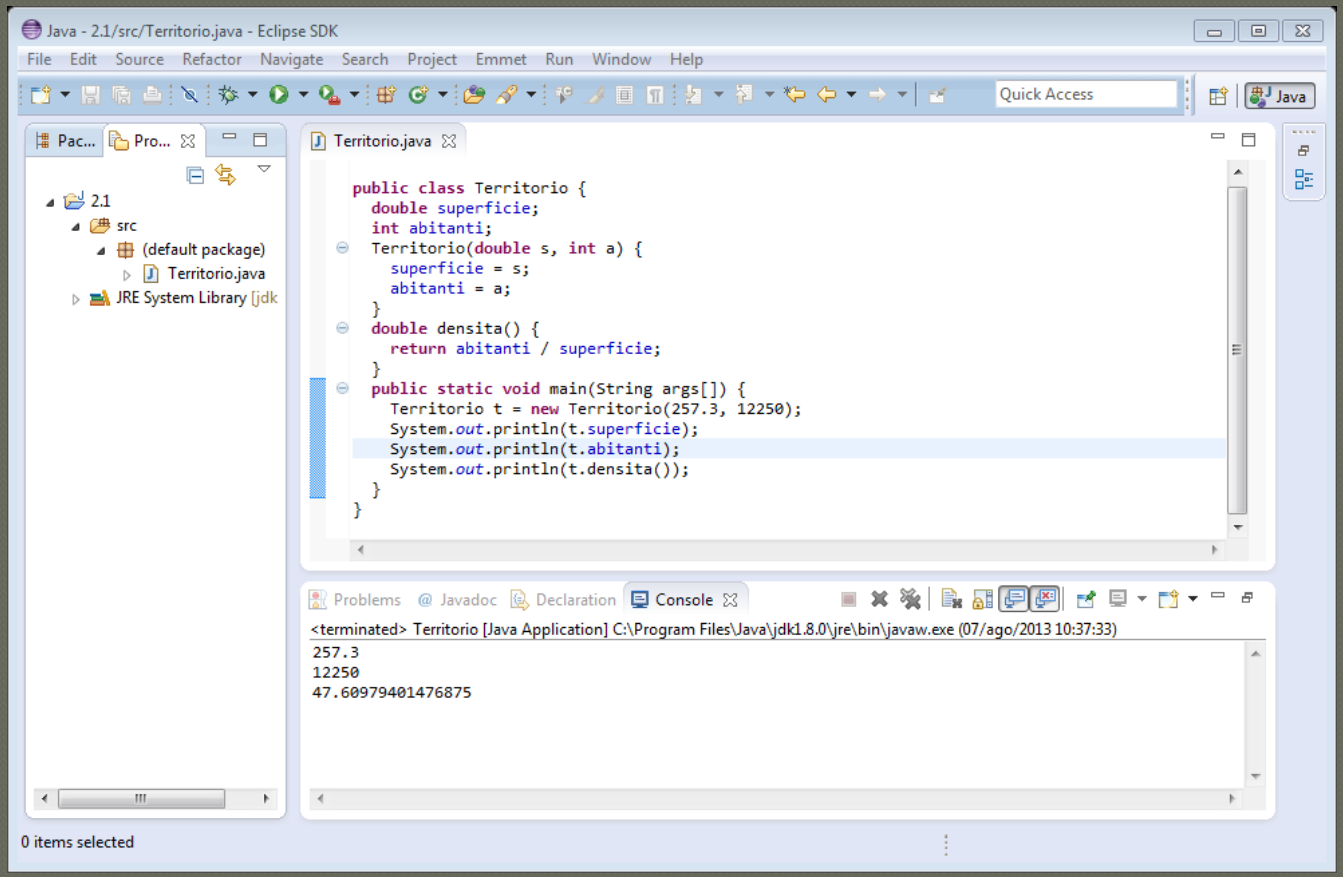
Codice

```
public class Territorio
{
double superficie;
int abitanti;
Territorio(double s, int a)
        {
        superficie = s;
        abitanti = a;
        }
        double densita ()
        {
        return abitanti / superficie;
        }
        public static void main(String args[])
        {
        Territorio t = new Territorio(257.3, 12250);
        System.out.println(t.superficie);
        System.out.println(t.abitanti);
        System.out.println(t.densita());
        }
}
```

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Eseguiamo il programma con eclipse

menu *Run* □ *Run (oppure: Ctrl+F11)*



Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Esercizi

1. Facendo riferimento al codice fornito nell'esempio della classe Territorio, individuare:
   - Dove vengono definite le variabili;
   - Dove vengono inizializzate le variabili;
   - Dove vengono definiti i costruttori e con quale nome;
   - Dove definiti i metodi e con quale nome, ne riconosce uno la cui importanza viene definita "critica", perché?
   - Dove vengono istanziate le classi;
   - Dove avvengono le operazioni di Input e Output;
   - Dove i metodi vengono applicati agli oggetti.

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Esercizi

2. Aggiungi al main() il codice per istanziare altri oggetti di classe Territorio, inizializzandoli con valori diversi dei campi. Verifica che ognuno di questi oggetti mantenga la sua copia delle variabili superficie e abitanti.
3. Aggiungi alla classe **Territorio un costruttore privo di** argomenti, che inizializza le variabili **superficie e abitanti con i valori rispettivamente 0.0 e 0 . Istanzia un** oggetto con tali valori. Che cosa succede se chiami il metodo **densita()?**

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Esercizi

4. Create una classe che contiene un int e un char non inizializzati, quindi visualizzatene i valori per verificare l'inizializzazione predefinita eseguita da Java.
5. Scrivere un programma che consenta di visualizzare a console, il valore di tre variabili: un numero intero, un numero reale e un booleano, dati in input da tastiera.

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Esercizi

6. **Si richiede l'anno di nascita e l'anno attuale e si fornisca in output l'età (Esercizio06.java)**
7. **Si richiede la misura della base e dell'altezza di un rettangolo e si fornisca in output il perimetro (Esercizio07.java)**
8. **Si richiede la misura dei cateti di un triangolo rettangolo e si fornisce in output la misura dell'ipotenusa (Esercizio08.java)**
9. **Si richiedono in input 3 valori e si fornisce in output la media aritmetica (Esercizio09.java)**

Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)