Struttura di un programma Java

Un programma in Java è un insieme di dichiarazioni di classi.
Una classe **non può** contenere direttamente delle **istruzioni**, **ma può** contenere
la dichiarazione di metodi,
che contengono **dichiarazioni** ed **istruzioni** (terminate da ;).

public class Main {
La classe Main contiene un metodo, che si chiama main.
In Java ogni metodo può essere chiamato (invocato) solo all'interno di un altro metodo. Quindi occorre un metodo iniziale per far partire l'esecuzione del programma.
Tale metodo iniziale è il metodo main, il quale deve essere sufficientemente generale.
L'intestazione del metodo main è
public static void main (String[] args)
ed è sempre la stessa.

Il metodo main si caratterizza per le seguenti proprietà:
- è **pubblico** (public), ovvero visibile da ogni altro punto del programma;
- è **statico** (static), in quanto all'inizio non sono stati creati ancora degli oggetti;
- non restituisce **niente** (void), in quanto non vi è alcun metodo che lo chiama a cui restituire un valore;
- i dati in **ingresso** sono visti come **array** di stringhe (oggetti della classe predefinita String).

Sappiamo che **da una classe** possiamo ottenere **molteplici istanze** e per ciascuna istanza si hanno variabili dai nomi identici ma dai valori distinti (forse "che puntano a locazioni di memoria diverse" sarebbe una definizione più chiara). Se poi vogliamo che una **variabile sia la medesima per tutte le istanze** di una classe sappiamo che la **dobbiamo invece definire come static**.
La **keyword static** in java viene usata per definire una **proprietà** di oggetti e **metodi** che sono **condivisi** da più istanze di una stessa classe.
Ciò significa che questo tipo di proprietà non è riferita ad una istanza della classe, ma bensì alla classe stessa. Infatti una modifica ad una variabile statica riflette su tutte le istanze di quella classe.

Per i metodi avviene sostanzialmente la medesima cosa:
- i **metodi non statici** sono associati ad ogni singola istanza di una classe e perciò il loro contesto di esecuzione (quindi l'insieme delle variabili cui possono accedere) è relativo all'istanza stessa.
- in contrapposizione i **metodi statici** non sono associati ad una istanza ma solo ad una classe. Quindi non potranno interagire con le variabili di istanza, ma solamente con quelle statiche.

void è una keyword ("parola chiave") ed è quindi una parola riservata del linguaggio che non può essere usata per altri scopi, In Java la parola chiave **void** in effetti **ha principalmente un solo ed unico uso**: dichiarare che un metodo non restituisce alcun valore.

Il parametro **args** rappresenta l'Array contenente tutti i parametri passati al programma (ad esempio dalla linea di comando).

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

L'operatore new serve a creare una nuova istanza di un oggetto appartenente ad una determinata classe e, conseguentemente, viene allocata automaticamente la memoria necessaria per conservare tale istanza. L'operatore new, per eseguire la creazione di un oggetto, invoca il costruttore della classe che si desidera istanziare.

```
        Array Elenco = new Array();

        Dichiaro le variabili necessarie
        final int NMAX = 10;                    //non può essere più assegnato
        int[] vett  = new int[NMAX];
        int[] vett1 = new int[NMAX];
        int[] vett2 = new int[NMAX*2];
        int Trovato;

        System.out.println("attraversa Array VUOTO ----------------------");
        Elenco.attraversaArray(vett);

        System.out.println("popola Array e visualizza --------------------");
        Elenco.popolaArray(vett,5,10);
        Elenco.attraversaArray(vett);
    }
}


class Array {
    // METODO popola Array
    public void popolaArray (int[] vett, int NumItem, int base) {
        for(int i=0; i<NumItem; i++)
            vett[i] = i*base;
    }

    // METODO attraversa Array
    public void attraversaArray (int[] vett) {
        for(int i=0; i<vett.length; i++)
            System.out.println(i + " " + vett[i]);
    }
}
```