Azzolini Riccardo 2018-11-06

Array e metodi

1 Argomento del metodo `main`

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

Il metodo `main` ha un argomento di tipo `String[]` (che per convenzione viene solitamente chiamato `args`): tramite tale array è possibile accedere agli argomenti passati a riga di comando all'avvio del programma.

1.1 Esempio

```
import prog.io.ConsoleOutputManager;

public class Ripeti {
    public static void main(String[] args) {
        ConsoleOutputManager out = new ConsoleOutputManager();

        for (int i = 0; i < args.length; i++) {
            out.println("args[" + i + "] = \"" + args[i] + "\"");
        }
    }
}
```

Esecuzione:

```
> java Ripeti abc 123
args[0] = "abc"
args[1] = "123"
```

2 Array multidimensionali

Un **array multidimensionale**, o **array di array**, è un array i cui elementi sono a loro volta array.

Dichiarazione di variabile e creazione di array bidimensionale:

```
tipo_base[][] nome_array = new tipo_base[num_righe][num_colonne];
```

In questo caso, tutte le righe hanno la stessa lunghezza. Questo tipo di array corrisponde effettivamente a una matrice. È anche consentito creare array in cui le righe possono avere lunghezze diverse:

```
tipo_base[][] nome_array = new tipo_base[num_righe][];

nome_array[0] = new tipo_base[num_colonne_0];
nome_array[1] = new tipo_base[num_colonne_1];
// ...
nome_array[num_righe - 1] = new tipo_base[num_colonne_n_1];
```

L'accesso agli elementi si effettua con la sintassi

```
nome_array[riga][colonna]
```

Le dimensioni sono accessibili tramite i campi `length` dell'array esterno e di ciascuna riga:

```
nome_array.length; // numero di righe
nome_array[i].length; // lunghezza della riga i
```

3 Sottoproblemi e sottoprogrammi

La scomposizione di problemi complessi in **sottoproblemi** è un meccanismo indispensabile per affrontarli.

Nella programmazione, essa corrisponde alla suddivisione di un programma in **sottoprogrammi**, che nel linguaggio Java si effettua tramite la scrittura di metodi.

3.1 Esempio

Come esempio di scrittura di metodi, si considera un programma che legge due vettori di interi della stessa lunghezza, ne calcola il vettore somma, quindi stampa tutti e tre i vettori. I principali passi che il programma deve svolgere sono:

1. leggere la lunghezza dei vettori
2. leggere il primo vettore
3. leggere il secondo vettore
4. calcolare il vettore somma
5. costruire le stringhe che rappresentano i tre vettori

6. stampare tali stringhe

Il codice corrispondente al programma, se scritto interamente nel metodo main, è:

```
// 1. lettura della lunghezza
int lunghezza = in.readInt("Lunghezza dei vettori? ");

// 2. lettura del primo vettore
out.println("Lettura primo vettore");
int[] vett1 = new int[lunghezza];
for (int i = 0; i < vett1.length; i++)
    vett1[i] = in.readInt("Elemento " + i + "? ");

// 3. lettura del secondo vettore
out.println("Lettura secondo vettore");
int[] vett2 = new int[lunghezza];
for (int i = 0; i < vett2.length; i++)
    vett2[i] = in.readInt("Elemento " + i + "? ");

// 4. calcolo della somma
int[] somma = new int[lunghezza];
for (int i = 0; i < somma.length; i++)
    somma[i] = vett1[i] + vett2[i];

// 5. costruzione delle stringhe
String strVett1 = "";
for (int i = 0; i < vett1.length; i++)
    strVett1 += vett1[i] + (i < vett1.length - 1 ? " " : "");
String strVett2 = "";
for (int i = 0; i < vett2.length; i++)
    strVett2 += vett2[i] + (i < vett2.length - 1 ? " " : "");
String strSomma = "";
for (int i = 0; i < somma.length; i++)
    strSomma += somma[i] + (i < somma.length - 1 ? " " : "");

// 6. stampa delle stringhe
out.println("Primo vettore: [" + strVett1 + "]");
out.println("Secondo vettore: [" + strVett2 + "]");
out.println("Vettore somma: [" + strSomma + "]");
```

Il codice al punto 2 è uguale a quello al punto 3, se non per il vettore su cui opera. In altre parole, i due blocchi di codice risolvono lo stesso problema ma operano su dati diversi. Lo stesso vale per la costruzione delle tre stringhe al punto 5.

Per eliminare queste ripetizioni, conviene scrivere dei metodi per la lettura di un vettore e per la costruzione della stringa corrispondente:

```
private static int[] leggiVettore(ConsoleInputManager input, int lung) {
    int[] vettore = new int[lung];
    for (int i = 0; i < vettore.length; i++)
        vettore[i] = input.readInt("Elemento " + i + "? ");
    return lung;
}

private static String generaStringa(int[] vettore) {
    String risultato = [];
    for (int i = 0; i < vettore.length; i++)
        risultato += vettore[i] + (i < vettore.length - 1 ? " " : "");
    return risultato;
}
```

Osservazioni:

- il modificatore `private` specifica che questi metodi sono utilizzabili solo all'interno della classe
- i parametri dei metodi (`input`, `lung` e `vettore`) sono vere e proprie variabili utilizzabili nel corpo dei rispettivi metodi
- i parametri e le *variabili locali* dichiarate nel corpo di un metodo sono locali (esistono solo all'interno del metodo), quindi i loro nomi si possono riutilizzare in altri metodi
- per restituire un risultato si usa l'istruzione `return`, seguita da un'espressione del tipo specificato nel prototipo del metodo
- quando un metodo viene invocato, gli argomenti vengono passati *per valore*: i valori di input (che possono valori di tipo primitivo o riferimenti a oggetti) vengono copiati nelle variabili corrispondenti ai parametri; nel caso dei tipi riferimento, ciò significa che le modifiche effettuate in un metodo su un oggetto ricevuto come argomento sono visibili anche all'esterno del metodo

Utilizzando questi metodi, è possibile semplificare notevolmente il codice del metodo `main`. Dato che gli altri metodi sono definiti nella stessa classe del `main`, all'interno di quest'ultimo è possibile invocarli utilizzando solo il nome (senza bisogno di indicare l'oggetto o la classe di appartenenza).

```
// 1. lettura della lunghezza
int lunghezza = in.readInt("Lunghezza dei vettori? ");

// 2. lettura del primo vettore
```

```
out.println("Lettura primo vettore");
int[] vett1 = leggiVettore(in, lunghezza);

// 3. lettura del secondo vettore
out.println("Lettura secondo vettore");
int[] vett2 = leggiVettore(in, lunghezza);

// 4. calcolo della somma
int[] somma = new int[lunghezza];
for (int i = 0; i < somma.length; i++)
    somma[i] = vett1[i] + vett2[i];

// 5. costruzione delle stringhe
String strVett1 = generaStringa(vett1);
String strVett2 = generaStringa(vett2);
String strSomma = generaStringa(somma);

// 6. stampa delle stringhe
out.println("Primo vettore: [" + strVett1 + "]");
out.println("Secondo vettore: [" + strVett2 + "]");
out.println("Vettore somma: [" + strSomma + "]");
```