Le basi del linguaggio Java

- Il controllo dell'esecuzione

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

La struttura di controllo
selezione

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

La dichiarazione di scelta

La dichiarazione **if-else** è il meccanismo principale per controllare il flusso di un programma. L'istruzione di selezione (if) permette l'esecuzione condizionale di un'istruzione, o la scelta tra due istruzioni, eseguendo o l'una o l'altra istruzione (non entrambe)
La sintassi per l'istruzione di selezione è la seguente:
primo caso:
if(espressione booleana)
istruzione
secondo caso:
if(espressione booleana)
istruzione
else
istruzione

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

La dichiarazione di scelta

L'espressione booleana deve produrre un risultato di tipo **boolean;** l'*istruzione* può essere semplice, oppure composta, cioè una serie di istruzioni, in questo caso vanno raggruppate all'interno di parentesi graffe.

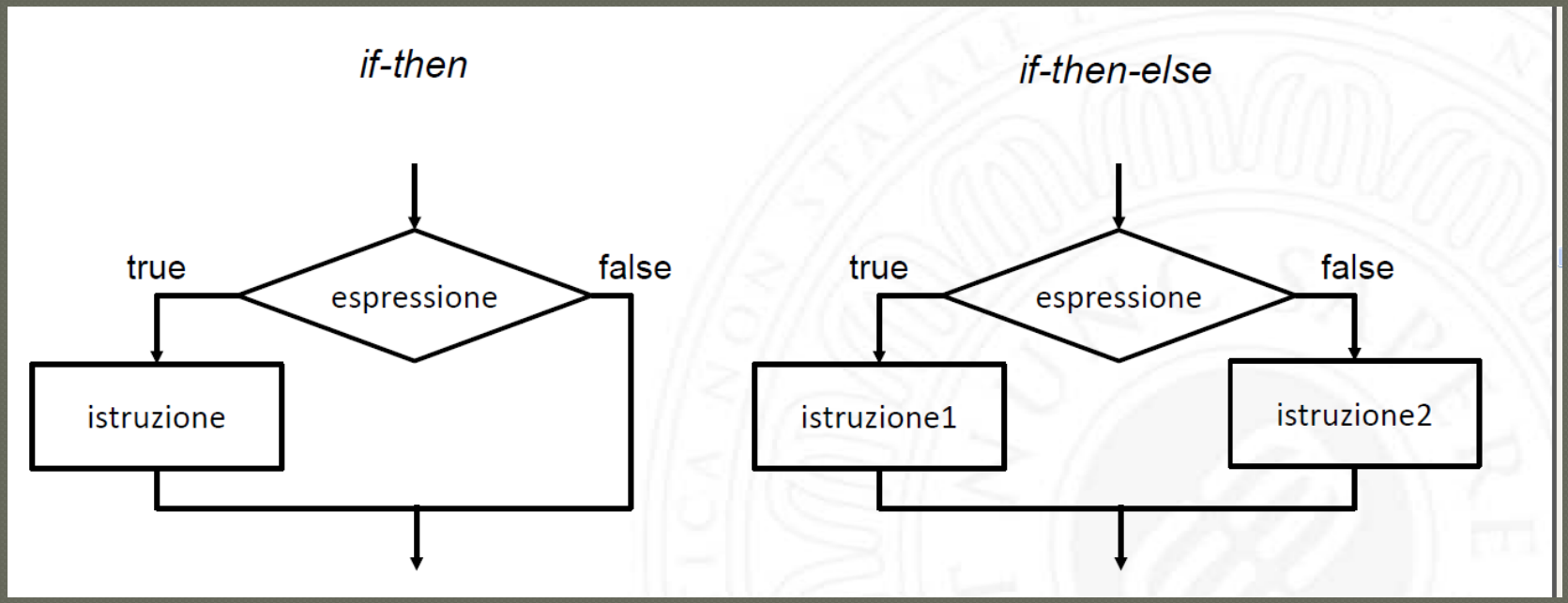
Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

La dichiarazione di scelta - esempio

```
public class IfElse {
static int result = 0;
static void test(int testval, int target) {
if(testval > target) // espressione booleana
{
result = +1;
}
else if(testval < target)
{
result = -1;
}
else
{
result = 0; // caso in cui i valori sono equivalenti
}
}
public static void main(String[] args) {
test(10, 5); // vengono passati i valori 10 e 5
System.out.println(result); //visualizza il risultato del test
test (5, 10);
System.out.println(result);
test(5, 5);
System.out.println(result);
}
}
```

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

Diagrammi di flusso dell'if



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

Gli operatori di confronto

L'operatore di uguaglianza è rappresentato da due simboli di uguale (==). Si faccia attenzione alla differenza rispetto all'operatore di assegnamento =, che ha un solo segno di uguale.

esempio:

```
if (voto == 6)
System.out.println("sufficiente");
```

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

Gli operatori di confronto

La disuguaglianza (diverso da) è espressa usando l'operatore !=
gli altri operatori di confronto sono:
<, <=, >, >=.

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

Gli operatori booleani

Gli operatori booleani vengono applicati a due valori booleani e restituiscono il risultato in base alle regole dell'algebra booleana.
L'operatore && indica l'operazione di AND.
L'operatore || indica l'operazione di OR.
L'operatore ! Indica la negazione.

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

Gli operatori booleani - esempio

```
if ((ora >= 12) && (ora <=17))
System.out.println("Buon pomeriggio");
else
System.out.println("Buona giornata");
```

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

La dichiarazione di scelta multipla

La struttura di selezione multipla consente di guidare l'esecuzione del programma scegliendo tra diverse alternative, a seconda del valore di una certa espressione.
L'espressione deve restituire un tipo intero oppure un carattere.

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

La dichiarazione di scelta multipla - esempio

```
switch (espressione)
{
case valore1:
// istruzione
break;
case valore2:
// istruzioni
break;
.....
.....
default:
// istruzioni
break;
}
```

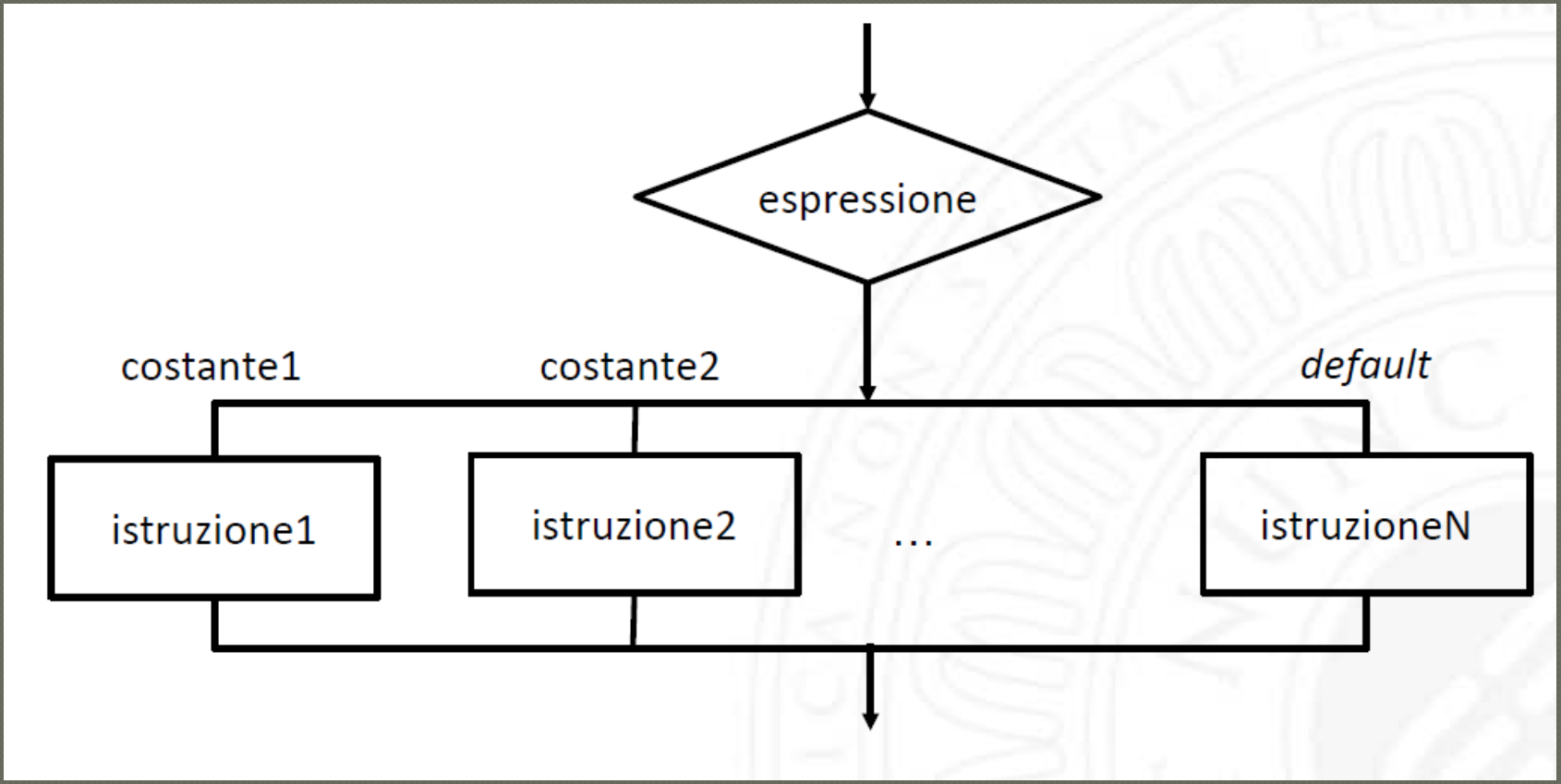
Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

La dichiarazione di scelta multipla

L'espressione dopo la parola chiave switch solitamente è una variabile di tipo intera. All'interno del blocco si elencano i possibili valori che può assumere l'espressione, usando per ogni valore un blocco case.
Durante l'esecuzione, a seconda del valore dell'espressione, viene eseguito il blocco case associato.
Se non viene trovato alcun valore nell'elenco dei case, si eseguono le istruzioni contenute nel blocco default.
La parola chiave break serve per indicare la fine di un blocco di istruzioni e fa terminare la selezione multipla.

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

Diagramma di flusso



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

La struttura di controllo
iterazione

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

Il ciclo while

Le istruzioni contenute nel blocco vengono eseguite mentre la condizione si mantiene vera. La condizione viene controllata prima di entrare nel ciclo, quindi se è falsa, le istruzioni non vengono eseguite nemmeno una volta.

sintassi:

```
While (condizione) // condizione deve essere di tipo boolean
{
// istruzioni
}
```

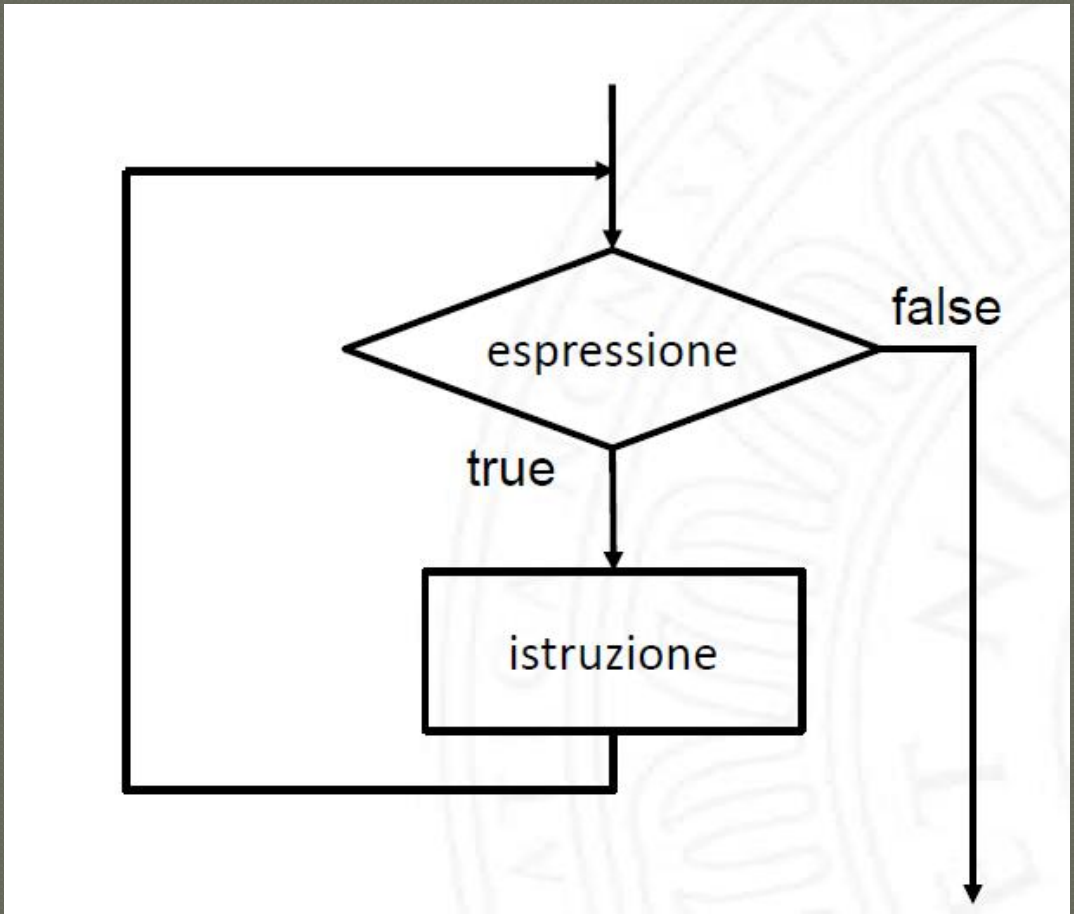
Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Il ciclo while - esempio

```
int i = 0;
while (i < 10) {
System.out.print(i);
i++;
}
```

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

Il ciclo while – diagramma di flusso



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

Il ciclo do while

L'istruzione do while ripete l'esecuzione di un'istruzione fin quando la condizione si mantiene vera.
Il controllo della condizione viene eseguito alla fine del blocco per cui le istruzioni vengono eseguite almeno una volta.
Sintassi:

```
do
{
// istruzioni
}
while (condizione);
```

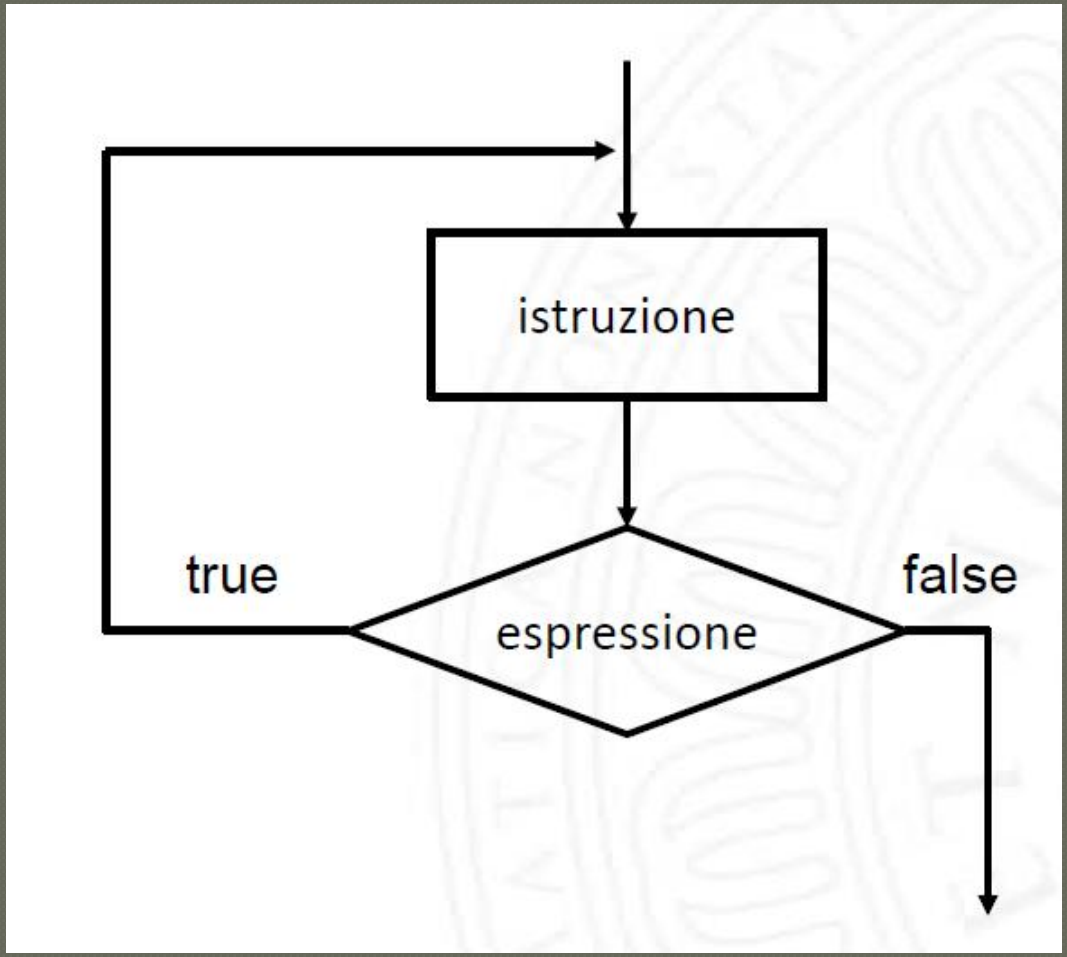
Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Il ciclo do while - esempio

```
int i = 0;
do {
System.out.print(i);
i++;
} while (i < 10);
```

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

Il ciclo do while – diagramma di flusso



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

Il ciclo for

Il ciclo for è la forma d'iterazione più utilizzata. Prima di eseguire la prima iterazione, questo ciclo esegue l'inizializzazione, poi esegue la verifica condizionale, e al termine di ogni iterazione "passo": incrementa o decrementa la variabile di controllo.

Sintassi:

```
For (inizializzazione; condizione; passo)
{
// istruzioni
}
```

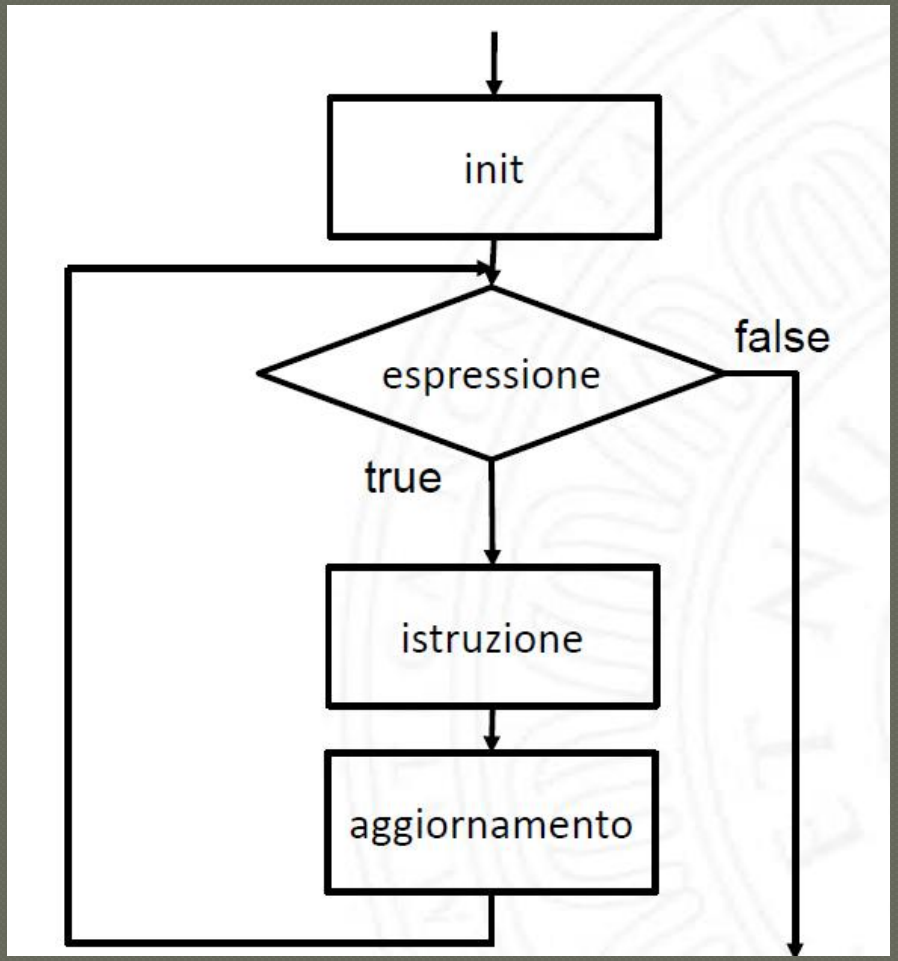
Autore: Prof. Agostino Sorbara
ITIS "M. M. Milano" Polistena (RC)

Il ciclo for - esempio

```
For(int i=1; i<=10; i++)
{
System.out.println(i);
}
```

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

Il ciclo for – diagramma di flusso



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

Il ciclo for – l'operatore virgola

L'operatore virgola, da non confondere con il separatore virgola utilizzato per separare le definizioni e gli argomenti dei metodi, ha solo un impiego in Java: nell'espressione di controllo di un ciclo **for**. Infatti, sia nell'inizializzazione sia nei passi dell'espressione di controllo possono essere presenti molte istruzioni, separate da virgole; tali istruzioni vengono valutate consecutivamente. Mediante l'operatore virgola possono essere definite diverse variabili all'interno di un ciclo **for**, purché siano dello stesso tipo.

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

Il ciclo for – l'operatore virgola - esempio

```
public class OperatoreVirgola
{
public static void main (String[] args)
{
for(int i =1, j = i + 10; i<5; i++, j=i*2)
{
System.out.println("i = " + i + " j = " + j);
}
}
}
```

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

La sintassi foreach

Per il ciclo **for** (a partire dalla versione SE5) java ha introdotto una sintassi nuova e più sintetica da utilizzare con array e contenitori. Prevede che non sia necessario creare un *int* per contare una sequenza di voci: **foreach** si incarica di controllare di produrre ogni voce automaticamente. Per esempio supponete di avere un array di float e di voler selezionare ogni elemento contenuto nell'array:

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

La sintassi foreach - esempio

```
import java.util.*;

public class ForEachFloat {
public static void main(String[] args) {
Random rand = new Random(47);
float f[] = new float[10];
for(int i = 0; i<10; i++)
f[i] = rand.nextFloat();
for(float x:f)
System.out.println(x);
}
}
```

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

La sintassi foreach

L'array è popolato utilizzando il "vecchio" ciclo **for**, poiché l'accesso ai suoi elementi deve avvenire mediante un indice. Nell'esempio for(float x:f)

In questo esempio viene definita una variabile x di tipo **float**, e ogni elemento di **f** è assegnato in sequenza a **x**.

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

La sintassi foreach

Qualsiasi metodo restituisca un array è un eccellente candidato all'utilizzo della sintassi **foreach**. Per esempio la classe String dispone del metodo **toCharArray()** che restituisce un array di **char**, e consente di iterare facilmente i caratteri presenti in una stringa.

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

La sintassi foreach - esempio

```
package forEachFloat;

public class ForEachString {
public static void main(String[] args) {
for(char c : "Un calabrese a Roma".toCharArray() )
System.out.print(c + " ");
}

}
```

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

La struttura di dati array

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

La struttura di dati array

Quando si devono memorizzare più oggetti che sono dello stesso tipo, si possono utilizzare gli **array** a una dimensione (vettori) o a più dimensioni (matrici).
Un array è realizzato con un riferimento che punta ad un area di memoria dove si trovano i suoi elementi.

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

La struttura di dati array

Per indicare l'array, cioè un gruppo di elementi dello stesso tipo, si usa un unico nome collettivo. Ogni elemento è individuato attraverso un indice (numero intero).
Il primo elemento dell'array ha come indice 0 (zero). *Il numero di elementi che compongono un array costituisce la dimensione dell'array. Quindi la posizione dell'ultimo elemento di un array è individuata dall'indice ottenuto meno uno.*

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

La struttura di dati array

Durante l'esecuzione del programma vengono controllati gli indici degli array e se non sono validi, cioè hanno superato i limiti consentiti, viene segnalato un errore.
Per poter creare e usare un array si devono seguire tre passaggi:

1. *Dichiarazione dell'array;*
2. *Allocazione;*
3. *Inizializzazione.*

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

Dichiarazione di un array

Consiste nella specificazione del suo tipo, può essere un tipo predefinito di Java (es. **int**, **float**, ecc.) oppure può essere una classe (per esempio **String**, **Automobile**). Si possono creare **array** che contengono un insieme di **oggetti** tutti della stessa **classe**.
Come per le variabili, la dichiarazione è fatta indicando un tipo e poi il nome da associare all'array. Però subito dopo il nome sono presenti le parentesi quadre, per indicare che è un array. Le parentesi possono essere messe anche dopo il tipo, ma la stragrande maggioranza dei programmatori utilizza le parentesi dopo il nome.

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

Dichiarazione di un array

Esempio:
int i [ ];
String nomi[ ];
L'array i è un array diinteri. L'array nomi è un array di oggetti String. Si osservi che la sola dichiarazione non specifica quale sia la dimensione dell'array. L'array a questo punto non è stato ancora creato e assume il valore speciale **null**.

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

Allocazione di un array

Consente di specificare quanto spazio riservare all'**array**, cioè quanti elementi dovranno essere memorizzati al suo interno. L'allocazione viene eseguita utilizzando l'operatore **new**, lo stesso operatore sarà utilizzato successivemente anche per allocare gli oggetti.

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

Allocazione di un array - esempio

L'allocazione delle variabili dell'esempio precedente viene fatta con le seguenti istruzioni:
i = new int [5];
nomi = new String[10];
L'operatore new è seguito dal tipo dell'array e dalla dimensione, specificata tra parentesi quadre. In questo caso è stato allocato l'array i per contenere 5 interi, e l'array nomi per contenere 10 oggetti String.
Dopo la fase di allocazione, si può dire che l'array è stato creato è può essere utilizzato, assegnando i valori ai suoi elementi.

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

Allocazione dinamica di un array

Il tipo di allocazione che abbiamo appena descritto viene definita di tipo statico, in quanto la sua dimensione viene definita mentre si crea il programma, tuttavia la dimensione dell'array può essere definita anche in modo dinamico, nel senso che la sua dimensione effettiva sarà stabilita quando il programma andrà in esecuzione e ne chiederà il valore, che al momento della creazione del programma viene definito con una variabile.

Esempio:

i = new int [a];

La variabile a rappresenta la dimensione dell'array che dovrà essere richiesto successivamente.

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

Inizializzazione di un array

Per assegnare i valori agli elemti di un array si procede nel seguente modo:

```
i[0] = 45;
i[1] = 30;
i[2] = 21;
.....
i[n-1] = 38;

nomi[0] = “Giulia”;
....
Nomi[n-1] = “Stefania”;
```

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

Gli array multidimesionali

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

Gli array multidimensionali

Gli array che abbiamo descritto finora corrispondono in matematica ai vettori.
Esistono altri tipi di array definiti multidimensionali, in particolare l'array bidimensionale, che corrisponde in termini matematici ad una matrice con righe e colonne, e che possiamo associare in generale ad una tabella. Ogni elemento della matrice è individuato da una coppia di numeri di cui il primo indica la riga ed il secondo la colonna.

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

Gli array multidimensionali - esempio

Di seguito viene proposto un esempio di una matrice con 2 righe e 3 colonne, matrice di dimensione 2x3.

0	2	3
2	4	5

In java la dichiarazione e l'allocazione di un array multidimensionale si realizza allo stesso modo della dichiarazione di un array monodimensionale: occorre specificare la nuova dimensione usando un secondo paio di parentesi quadre.
Esempio:

```
int matrice[ ] [ ];
matrice = new int [2] [3];
```

Per assegnare i valori alla matrice bisogna riferirsi a uno specifico elemento della matrice indicando la sua posizione come una coppia di indici, il seguente esempio assegna il valore 5 al primo elemento della matrice.

```
matrice [0] [0] = 5;
```

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