Lab. Programmazione (CdL Informatica) & Informatica (CdL Matematica) a.a. 2022-23

Monica Nesi

Università degli Studi dell'Aquila

26 Ottobre 2022

Array in Java

Costrutto per rappresentare *sequenze di elementi dello stesso tipo*.

Si dice che l'array è una *struttura dati omogenea* (elementi tutti dello stesso tipo).

Il costrutto array [] può essere visto come un *costruttore di tipo*.

Dato un *qualsiasi* tipo T, applicare il costruttore di tipo [] risulta nel nuovo tipo T[], ovvero il tipo *array (monodimensionale)* con elementi di tipo T.

Esempio: `int[]` è il tipo *array di interi*, i cui elementi sono sequenze lineari (monodimensionali) di interi.

Analogamente per `char[]`, `boolean[]`, `String[]`, etc.

Array in Java: lunghezza

Gli array hanno una *lunghezza* (o dimensione) *finita*.

La lunghezza di un array è il numero di elementi che l'array contiene.

La lunghezza di un array monodimensionale deve essere *fissata* in fase di *creazione* dell'array.

La lunghezza *non influenza il tipo*, e.g. un array di 5 interi ha lo stesso tipo di un array di 10 interi, ovvero `int[]`.

Un array monodimensionale (detto anche *vettore*) di lunghezza m può essere visto come un insieme di m variabili dello stesso tipo, caratterizzate dallo *stesso nome* e distinguibili l'una dall'altra tramite un *indice*, che può variare tra 0 ed $m-1$.

Array in Java: dichiarazione e creazione

Prima di creare un array occorre *dichiararlo*:

```
int[] a;
```

Questa dichiarazione dice che l'identificatore `a` è di tipo `int[]`, ovvero lo si vuole usare per *riferire* un array di interi (che ancora non è stato creato).

Per creare l'array (e quindi allocare lo spazio per contenere i suoi elementi) si usa la parola riservata `new` seguita dal tipo degli elementi e dalla lunghezza dell'array racchiusa dentro le parentesi quadrate `[]`:

```
a = new int[5];
```

L'operatore `new` alloca lo spazio necessario ed inizializza gli elementi dell'array.

Array: oggetti in Java

Gli array in Java sono implementati come particolari *oggetti*.

Essendo oggetti, si ha che:

- quando vengono dichiarati, gli array hanno una inizializzazione di default al valore convenzionale `null`;

- quando vengono creati, i loro elementi hanno una inizializzazione di default al valore di default del tipo degli elementi.

```
int[] a;
a = new int[5];
```

Dopo la dichiarazione di a, il suo valore è `null`.

Dopo la creazione dell'array a con `new`, i suoi elementi valgono 0.[1]

[1] Il valore di default per i tipi numerici è 0, per i booleani è `false`, per i caratteri è la codifica binaria con tutti 0 (ovvero il *carattere nullo*, che non è stampabile), per gli oggetti è il valore `null`.

Elementi di un array

Un elemento di un array è individuato tramite l'*identificatore* dell'array ed un *indice* (più in generale, un'espressione numerica) racchiuso tra `[]`, il cui valore deve essere compreso tra 0 e la lunghezza dell'array *meno* un'unità.

Dato l'array a di lunghezza 5, i suoi elementi sono individuati tramite espressioni del tipo `a[e]`, dove il valore di e deve variare tra 0 e 4 (altrimenti si ha errore).

N.B. L'indice con cui indichiamo gli elementi di un array parte da 0 e NON da 1. Quindi, il primo elemento dell'array è quello in posizione 0, il secondo è in posizione 1, etc.

Se *m* è la lunghezza di un array, l'elemento in posizione *m* NON esiste (si è già *fuori dall'array*).

Elementi di un array (cont.)

Ad esempio, possiamo assegnare dei valori agli elementi dell'array a:

```
a[0] = 3;
a[2] = 6;
a[4] = a[0]+1;
```

Se poi stampo i suoi elementi con

```
for (int i=0; i<5; i++)
  System.out.println(a[i]);
```

si ottiene

```
3
0
6
0
4
```

Ancora dichiarazione e creazione di array

È possibile dichiarare e creare un array con una sola riga di codice:

```
int[] b = new int[5];
```

secondo la sintassi

$$<Tipo>[\,]\ <Ide> = \ new\ <Tipo>\ [<Espr>]\,;$$

dove il tipo deve essere lo stesso e l'espressione deve essere numerica.

N.B. La lunghezza dell'array deve essere *sempre* fissata in fase di creazione dell'array (altrimenti si ha errore in compilazione).

```
for (int i=0; i<5; i++)
  b[i] = i+1;
```

assegna agli elementi dell'array `b` i valori da 1 a 5.

Creazione per enumerazione

Quando si conoscono gli elementi di un array, è possibile crearlo *enumerando* gli elementi nell'ordine, racchiusi tra parentesi graffe e separati da virgole, *senza* usare `new` e *senza* indicare la lunghezza dell'array:

```
int[] b = {1,2,3,4,5};
```

La lunghezza dell'array (che in questo caso viene ricavata automaticamente da Java) può essere sempre ottenuta scrivendo il nome dell'array seguito da un punto . e dalla parola `length`:

```
int m = b.length;
```

assegna 5 alla variabile intera `m`.

`length` è una *variabile istanza* dell'oggetto array.

N.B. La lunghezza `a.length` di un qualsiasi array a NON è modificabile!

Assegnamenti tra array

Una variabile di tipo array può essere assegnata ad un'altra dello stesso tipo.

N.B. Una variabile o identificatore di tipo array è solo una *referenza* (un *riferimento*, *puntatore*) ad una zona di memoria.

Sia dato il frammento di codice:

```
int[] b = new int[5];
for (int i=0; i<b.length; i++)
  b[i] = i*2;
int[] a = b;     // assegnamento tra array
a[2] = 5;        // modifica l'array
b = null;
```

Alla fine l'identificatore b non riferisce più l'array, riferito solo dall'identificatore a.

Test su array

```
class TestArray {
  public static void main (String[] args) {
    int[] a = new int[3];
    a[0] = 2;
    a[1] = -3;
    a[2] = 1;
    System.out.println(a.length);
    int[] b = {11,3,-7,6};
    a = b;
    System.out.println(a.length);
    System.out.println(a[3]);
    int[] c = {-5,8};
    b = c;
    System.out.println(b.length);
    System.out.println(b[2]);
  }
}
```

Test su array: output

Verificare che l'output del programma precedente è il seguente:

```
3
4
6
2
Exception in thread "main"
java.lang.ArrayIndexOutOfBoundsException: 2
        at TestArray.main(TestArray.java:15)
```

... e cercare di capire perché.