

# Note ed esercizi aggiuntivi

*Gli esercizi proposti sono utili per rivedere gli esempi riportati, che sono stati sviluppati e discussi in dettaglio a lezione.*

## 8. Array

*Esempio.* Lettura di una sequenza di stringhe terminata dalla stringa vuota (che non fa parte della sequenza) e riscrittura sul monitor della sequenza stessa.

```
import prog.io.ConsoleInputManager;
import prog.io.ConsoleOutputManager;

class SequenzaStringhe {

    public static void main(String[] a) {
        ConsoleInputManager in = new ConsoleInputManager();
        ConsoleOutputManager out = new ConsoleOutputManager();

        final int MAX = 10; //capacita' dell'array
        String[] tab = new String[MAX];

        //fase di lettura
        int pos = 0; //prima posizione libera
        String x = in.readLine("Stringa? ");
        while (!x.equals("")) {
            tab[pos++] = x;
            x = in.readLine("Stringa? ");
        }

        //fase di scrittura
        for (int i = 0; i < pos; i++)
            out.println(tab[i]);
    }
}
```

*Note*

- Si osservi l'uso dell'operatore di *incremento postfisso* per accedere alla posizione `pos` dell'array, cui viene assegnata la stringa letta, e per predisporre, allo stesso tempo, `pos` sulla posizione successiva.

- Se l'utente inserisce più di 10 stringhe (costante `MAX`) l'esecuzione dell'applicazione viene interrotta dal tentativo di accedere a una posizione dell'array non esistente. Provate a far eseguire l'applicazione per vedere il messaggio di errore che viene visualizzato.

Nella versione seguente la lettura delle stringhe viene terminata nel caso l'array sia completamente riempito.

```
import prog.io.ConsoleInputManager;
import prog.io.ConsoleOutputManager;

class SequenzaStringhe {

    public static void main(String[] a) {
        ConsoleInputManager in = new ConsoleInputManager();
        ConsoleOutputManager out = new ConsoleOutputManager();

        final int MAX = 10; //capacita' dell'array
        String[] tab = new String[MAX];

        //fase di lettura
        int pos = 0; //prima posizione libera
        String x = in.readLine("Stringa? ");
        while (!x.equals("") && pos < MAX) {
            tab[pos++] = x;
            if (pos < MAX)
                x = in.readLine("Stringa? ");
        }

        //fase di scrittura
        for (int i = 0; i < pos; i++)
            out.println(tab[i]);
    }
}
```

#### *Note*

La capacità dell'oggetto array è stabilita al momento della creazione (invocazione del costruttore `new String[...]`). Ciò limita l'utilità di questo codice: il ciclo di lettura termina comunque quando si raggiunge la capacità. Si potrebbe leggere preliminarmente il numero di stringhe da inserire, prima di creare l'oggetto array, in modo da dimensionarlo opportunamente. Tuttavia questo richiede all'utente di sapere o contare a priori le stringhe da inserire, soluzione del tutto insoddisfacente. Una soluzione migliore è proposta nell'esercizio successivo.

#### **Esercizio 8.1**

Fate riferimento all'esempio precedente. Quando l'array risulta pieno, è possibile simularne l'ampliamento come segue.

- Si crea un nuovo oggetto array (raggiungibile tramite un riferimento ausiliario) di capacità maggiore, ad esempio doppia.
- Si copiano tutti gli elementi presenti nell'array riferito da `tab` nelle corrispondenti posizioni del nuovo array.
- Si copia in `tab` il riferimento al nuovo array.

A questo punto si può proseguire l'esecuzione leggendo una nuova stringa.

Queste operazioni possono essere codificate all'interno del ciclo `while` in modo che siano effettuate ogni volta che si raggiunge la capacità dell'array che si sta utilizzando, cioè ogni volta che `pos` risulti uguale a `tab.length`. Effettuate queste modifiche sul codice precedente.

*Nota.* Classi come `Vector` o `ArrayList` delle librerie standard di Java utilizzano proprio un meccanismo di questo tipo per simulare array la cui capacità può variare dinamicamente.

*Esempio.* Visualizzazione del numero totale di argomenti forniti sulla linea di comando, della media delle loro lunghezze. Gli argomenti vengono poi visualizzati sul monitor, trasformando tutte le lettere minuscole in maiuscole.

```
class Argomenti {
    public static void main(String[] a) {
        int nArgs = a.length;
        System.out.println("Numero argomenti: " + nArgs);

        int sommaLunghezze = 0;
        for (String s: a)
            sommaLunghezze += s.length();
        double mediaLunghezze = (double) sommaLunghezze / nArgs;

        System.out.println("MediaLunghezze: " + mediaLunghezze);

        for (String s: a)
            System.out.println(s.toUpperCase());
    }
}
```

*Note*

Nel caso sulla linea di comando non si forniscano argomenti, la variabile `a` si riferisce a un array di lunghezza zero. Si faccia attenzione alla differenza tra array vuoto (l'array c'è ma non ha elementi, pertanto il campo `length` contiene 0) e il riferimento `null` (l'array non esiste, e dunque non esiste nemmeno il campo `length`).

### Esercizio 8.2

Se non si forniscono argomenti, come media viene visualizzato `NaN`, *Not-a-Number*, risultato della divisione di 0 per 0 nel tipo `double`. Modificate il codice in modo che in questo caso, anziché visualizzare la media, visualizzi un messaggio che indichi che non sono stati forniti argomenti. Scrivete due soluzioni:

1. nella prima introducete una selezione che eviti di effettuare la divisione se il numero di argomenti è zero;
2. nell'altra soluzione effettuate la divisione e poi controllate se il risultato è `NaN` per decidere cosa visualizzare. La classe involucro `Double` fornisce un campo statico di nome `NaN` il cui valore rappresenta proprio il valore `NaN` di tipo `double`.

*Esempio.* L'applicazione dell'esempio precedente viene richiamata da linea di comando scrivendo

```
java Argomenti
```

seguito, eventualmente, dagli argomenti che verranno collocati dalla Java Virtual Machine nell'array riferito dal parametro `a`. Nulla vieta che un'altra classe richiami il metodo statico `main` della classe `Argomenti`, come fa ad esempio l'applicazione seguente (è necessario che le due applicazioni siano nella stessa directory):

```
class UsaArgomenti {
    public static void main(String[] a) {

        String[] arrayTest = new String[3];
        arrayTest[0] = "cane";
        arrayTest[1] = "gatto";
        arrayTest[2] = "topo";
        Argomenti.main(arrayTest);

        arrayTest = new String[0];
        Argomenti.main(arrayTest); //argomento: array vuoto

        arrayTest = null;
        Argomenti.main(arrayTest); //argomento: riferimento null
                                   //errore durante l'esecuzione di Argomenti.main
    }
}
```