LABORATORIO DI INFORMATICA 1

2014-2015

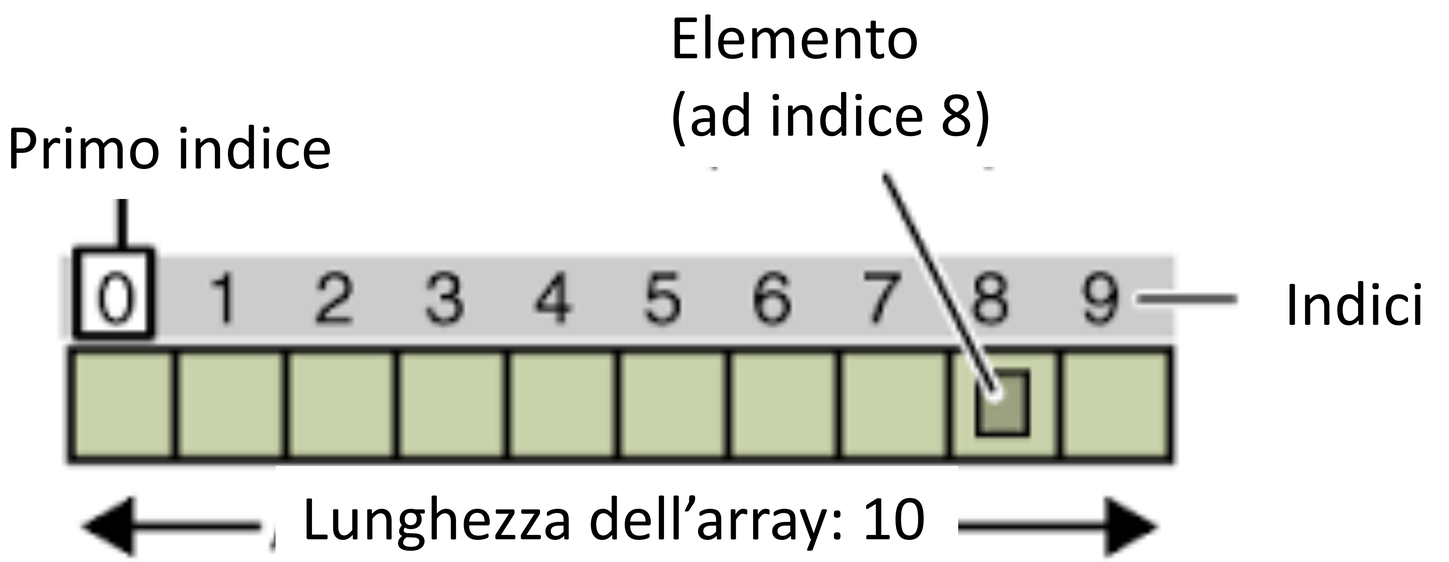
- parte 3 -

Array

Se dovessimo salvare le temperature di ciascun giorno in un anno, utilizzeremmo 365 variabili di tipo intero?

Per fortuna no, ci sono gli ARRAY!

Un array (o vettore) è una struttura dati che contiene un numero finito di elementi (tutti dello stesso tipo)



Array

Dichiarazione

tipo **nome** **[ ]** ; oppure **tipo** **[ ]** **nome**;

Inizializzazione

nome = new tipo [dimensione];

Dichiarazione e inizializzazione

tipo [ ] nome = new tipo [dimensione];

oppure

tipo nome [ ] = new tipo [dimensione];

Array

Esempio:

Dichiarazione

```
int [ ]   vettore;
```

Inizializzazione

```
vettore  =  new  int [ 10 ] ;
```

Dichiarazione e inizializzazione

```
int [ ]   vettore  =  new  int [ 10 ];
```

oppure

```
int       vettore [ ] =  new  int [ 10 ];
```

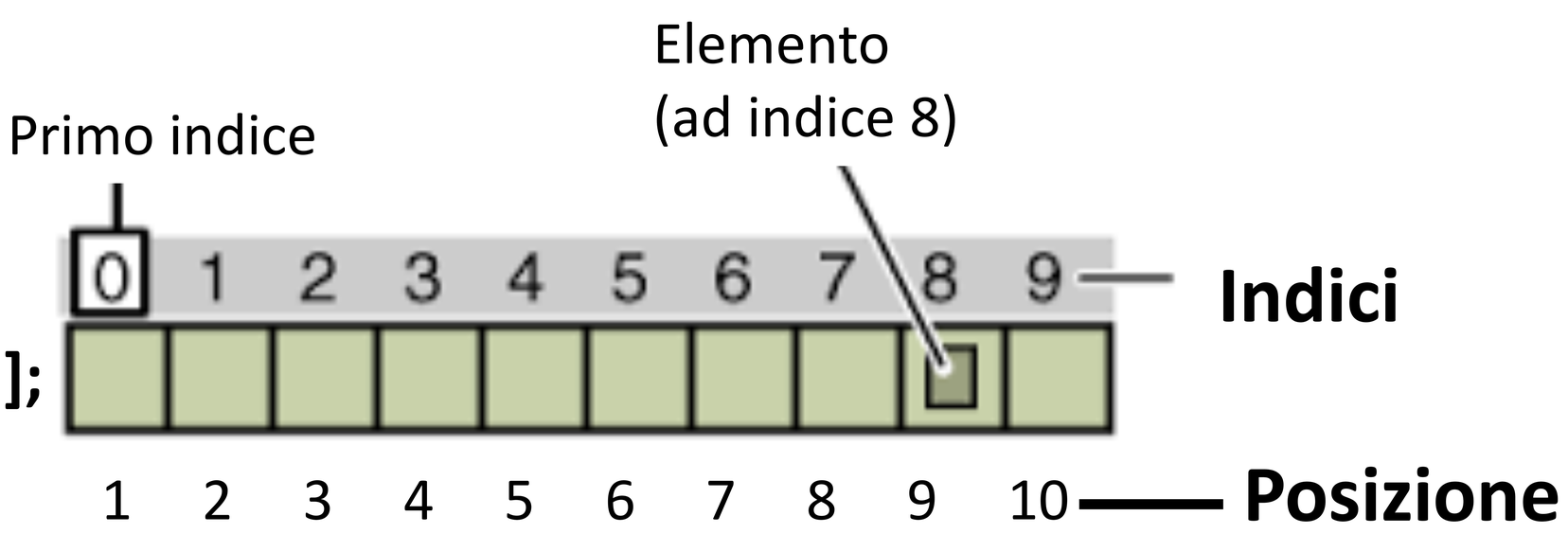
Array

Ciascun elemento dell'array viene identificato con un numero, chiamata ***indice***, che vale la **posizione – 1**

L'elemento in testa all'array, in **posizione 1**
ha indice **0**

L'elemento in coda all'array, in **posizione uguale alla dimensione dell'array**
ha indice **numero degli elementi meno uno**

```
int [ ] vettore = new int [ 10 ];
```



Array

Per accedere ad un elemento dell'array si usa la sintassi:

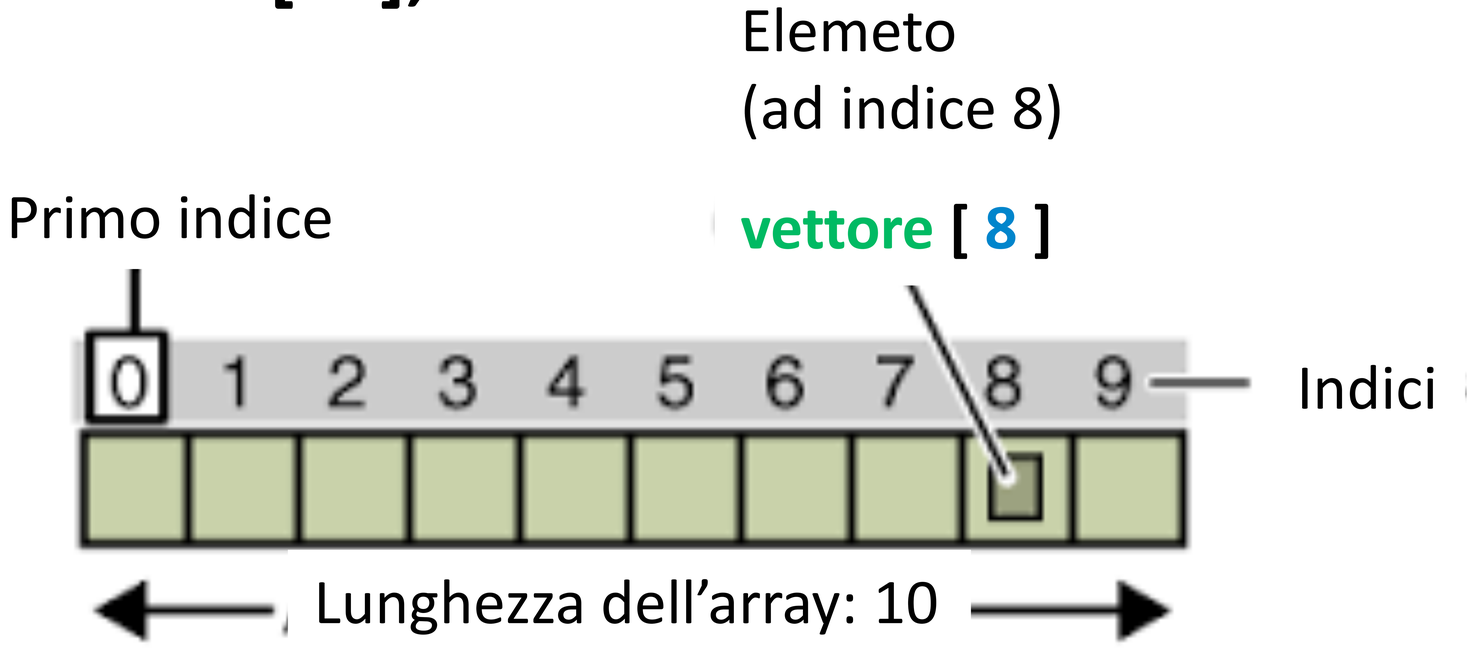
```
nomeArray [ indice ];
```

Dato

```
int [ ]   vettore   =   new   int [ 10 ];
```

Per accedere in posizione 8 come mostrato in figura, la sintassi è:

```
vettore [ 8 ];
```



Array

Appena creato un array è “vuoto”:

un array di interi	ha tutti gli elementi uguali a	0
un array di double	ha tutti gli elementi uguali a	0.0
un array di booleani	ha tutti gli elementi uguali a	false
un array di stringhe	ha tutti gli elementi uguali a	“” (stringa vuota)

Per assegnare un valore ad un elemento di un array:

vettore [posizione] = valore;

Esercizio

1. Inizializzare un array di interi e inserire in ciascun elemento il valore 5 [utilizzare un ciclo for]
2. Inizializzare un array di interi e assegnare a ciascun elemento un valore uguale alla sua posizione [utilizzare un ciclo for]
3. Dopo aver inizializzato l'array precedente, copiare il suo contenuto in ordine inverso in un altro array della stessa dimensione

Suggerimento:
la lunghezza di un array si ottiene con

nomeArray.length

L'ultimo elemento di un array è quindi

nomeArray [nomeArray.length - 1]

Il ciclo deve terminare dopo che si è acceduto a questo elemento

Procedure

Utilizzate per rendere il codice più:

- **Leggibile**
- **Riutilizzabile (la stessa funzione può essere invocata più volta)**

Questa è la **DEFINIZIONE** di una procedura

visibilità della procedura

(non ci interessa, usatela sempre così)

indica una procedura

```
public static void nomeProcedura( parametri formali ) {
    comandi
}
```

Procedure (in questa slide le procedure NON hanno parametri formali)

```
1
2  public class EsempioProcedura {
3
4      public static void stampaCiao() {
5          System.out.println("Ciao");
6      }
7
8      public static void stampaCiaoSuPiuRighe() {
9          System.out.println("C");
10         System.out.println("I");
11         System.out.println("A");
12         System.out.println("O");
13     }
14
15     public static void main(String[] args) {
16         stampaCiao();
17         stampaCiaoSuPiuRighe();
18     }
19 }
```

Chiamate alle procedure

Procedure

Parametri formali
(il nome è solo un caso sia lo stesso)

```
public class EsempioProcedura {

    public static void stampaNumero(int numero) {

        System.out.println("Numero vale: " + numero);

    }

    public static void main(String[] args) {

        int numero = 10;

        stampaNumero( numero );

    }
}
```

Chiamate alle procedura

Parametri in input alla funzione
(detti parametri attuali)

Funzioni

Sono PROCEDURE che restituiscono un valore

Questa è la **DEFINIZIONE** di una funzione

visibilità della funzione
(non ci interessa, usatela sempre così)

indica il tipo di ritorno della funzione

public static **tipoRiTORNO** **nomeFunzione (parametri formali) {**

comandi

}

Funzioni

Tipo di ritorno della funzione

```
public class EsempioProcedura {

    public static int moltiplica(int x, int y) {

        System.out.println("Moltiplico " + x + " e " + y);

        return x * y;

    }

    public static void main(String[] args) {

        int x = 10;
        int y = 20;

        int moltiplicazione = moltiplica(x, y);

        System.out.println("La moltiplicazione vale: " + moltiplicazione);

    }
}
```

Cosa ritorno
(la parola return indica quello che viene ritornato dalla funzione)

Chiamata alla funzione

Esercizio

1. Scrivere un programma in cui viene invocata una funzione che calcoli il fattoriale di un numero e successivamente (quando il valore viene ritornato) viene invocata una procedura che stampi il numero scelto e il valore del suo fattoriale

2. Delegare ad una funzione il compito di calcolare la somma di tutti i numeri da 1 a n

3. Inizializzare un array di double con valori random (utilizzate la funzione Math.random() che genera un valore compreso tra 0 e 1). Successivamente scrivete una funzione che cerchi il valore minimo dell'array e lo ritorni. Quindi stampate il valore.

4. Fate lo stesso per il valore massimo dell'array

Iterazione e ricorsione

Una funzione si dice **RICORSIVA** quando **RICHIAMA SE STESSA**
Se non lo fa allora è **ITERATIVA**

Quelle viste nelle precedenti slide e negli esercizi sono funzioni iterative

Per la ricorsione dobbiamo trovare i vari casi…
Per il fattoriale (ad esempio) abbiamo due casi:

- 1 se n = 0 (caso base)
- n * fattoriale (n - 1) se n > 0 (caso ricorsivo)

Ricorsione

```
public static int fattoriale (int numero) {

        if ( numero == 0 ) {
                return 1;
        } else {
                return numero * fattoriale ( numero - 1);
        }

}
```

Esercizio

1. Scrivere una funzione ricorsiva che calcoli la somma di tutti i numeri da 1 a n;

2. Scrivere un programma che dato un array di double (creato da voi), lo scorra tramite una funzione e tramite un'altra funzione stampi se quel numero appartiene all'area sottesa di $f(x) = x^2$
   - Nel main create un array e assegnategli dei valori random (usando Math.random())
   - Create una funzione che scorra questo array
   - Create una funzione che, dato in input un double, stampi il numero ricevuto in input e se appartiene all'area sottesa di f(x)
   - in questo caso le funzioni sono iterative

3. Scrivere una funzione ricorsiva che calcoli la sequenza di fibonacci

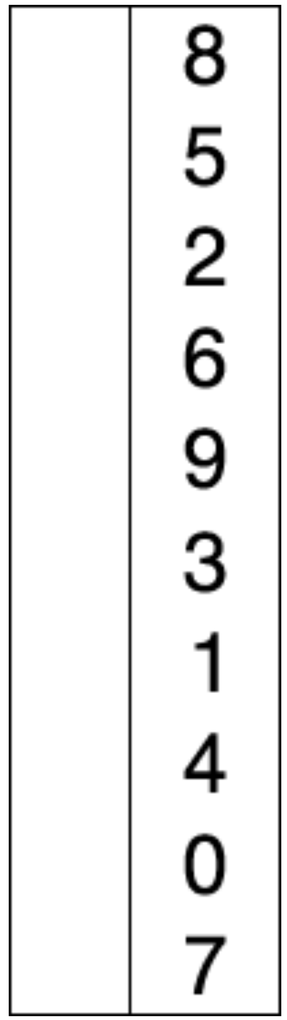
Ordinamento di un array

Tantissime implementazioni

Algoritmo base:

Dovendo ordinare un array *A* di lunghezza *n*, si fa scorrere l'indice *i* da 1 a *n-1* ripetendo i seguenti passi:

- si cerca il più piccolo elemento della sottosequenza *A*[*i*..*n*];
- si scambia questo elemento con l'elemento *i*-esimo.



http://disi.unitn.it/~foroni/teaching/mat/Selection-Sort-Animation.gif

Array

A differenza delle altre variabili, se un array viene modificato in una funzione resta modificato per tutto il programma.

Es:
```
double[] array = new double[2];
array[0] = 1; array[1] = 0;
ordinaVettore(array);   // assumiamo che questa funzione ordini un array
stampaVettore(array); // l'array stampato sarà ordinato
```

Con una variabile "normale" (non un array), se modifico un valore in una funzione all'esterno della funzione il valore resta uguale

Es:
```
int esempio = 10;
cambiaEsempio(esempio);          //assumiamo che la funzione cambiaEsempio assegni alla variabile in input 12
System.out.println(esempio);     //il valore stampato è 10, non 12!
```

Variabili Globali e Locali

```
public class EsempioProcedura {

    static int numero = 2;

    public static void stampaNumero() {

        System.out.println("Numero vale: " + numero);

    }

    public static void main(String[] args) {

        int k = 12;

        System.out.println("K vale: " + k);

        stampaNumero();

        numero = 10;

        stampaNumero();

    }
}
```

La variabile *numero* è **GLOBALE**: cioè è visibile (usabile) in **tutto** il programma

La variabile k è **LOCALE**: cioè è visibile (usabile) solo **all'interno del metodo in cui è stata dichiarata** (in questo caso solo all'interno del metodo main)

Qui viene stampato "Numero vale: 2"

Qui viene stampato "Numero vale: 10"

Variabili Globali e Locali

```
public class EsempioProcedura {

    static int numero = 2;

    public static void stampaNumero() {

        System.out.println("Sono in metodo stampaNumero() e numero vale: " + numero);

    }

    public static void stampaNumero(int numero) {

        System.out.println("Sono in metodo stampaNumero(int numero) e numero vale: " + numero);

    }


    public static void main(String[] args) {
        int numero = 10;

        System.out.println("Sono in metodo main e numero vale: " + numero);

        stampaNumero();

        stampaNumero(numero);
    }
}
```