Contenitori in Java

- Array
- Vettori (ArrayList)
- Liste (LinkedList)
- Iteratori
- Altri contenitori

Array in Java

- In Java, come in altri linguaggi è definito, il tipo array
- Un array è una sequenza di elementi dello stesso tipo
- Il numero di elementi costituisce la lunghezza dell'array
- La lunghezza viene fissata al momento della creazione
- La lunghezza di un array non è modificabile

Array: un esempio

```
...
Employee[] Dipendenti = new Employee[3];

Dipendenti[0] = new Employee("Mario", "Rossi");
Dipendenti[1] = new Employee("Ada", "Verdi");
Dipendenti[2] = new Employee("Leo", "Bianchi");

for(int i=0; i< Dipendenti.length; i++)
      System.out.print(Dipendenti[i].getFirstName() + "\n");

Dipendenti[3] = ....; //genera errore in fase di esecuzione

for( Employee d : Dipendenti)  //ciclo generalizzato
       System.out.print(d.getLastName() + "\n");
```

length: attributo costante non modificabile

Vettori in Java

- In Java è definita una classe parametrica ArrayList
- Un ArrayList consente di memorizzare una raccolta di oggetti
- Un ArrayList si comporta come un array ma:
  - Può crescere e decresce in base alle necessità
  - Ha dei metodi per svolgere le funzioni più comuni come l'inserimento e la rimozione degli elementi

ArrayList: un esempio

```
...
ArrayList<Employee> dipendenti = new ArrayList<Employee>();

dipendenti.add(new Employee("Mario", "Rossi"));
dipendenti.add(new Employee("Ada", "Verdi"));
dipendenti.add(0, new Employee("Leo", "Bianchi"));

for (Employee e: dipendenti)
        System.out.print(e.getFirstName()+"\n");

int i= dipendenti.size();
dipendenti.get(i); //errore in fase di esecuzione
Employee e1= dipendenti.get(1); //lettura
dipendenti.set(2,e1);                  //scrittura
dipendenti.remove(1);                  //rimozione

ArrayList<int> a = new ArrayList<int>(); //errore: int è un tipo base
```

for generico

Liste in Java

- Le liste sono realizzate mediante la classe parametrica LinkedList
- Una LinkedList consente di memorizzare una raccolta di oggetti mediante elementi concatenati
- Una LinkedList ha una interfaccia simile ad un ArrayList ma:
  - L'inserimento di nuovi elementi è generalmente più veloce
  - L'accesso agli elementi è sequenziale (generalmente più lento)

LinkedList: un esempio

```
...
LinkedList<Employee> dipendenti = new LinekdList<Employee>();

dipendenti.addFirst(new Employee("Mario", "Rossi")); //in testa
dipendenti.addLast(new Employee("Ada", "Verdi"));    //in coda
dipendenti.add(1, new Employee("Leo", "Bianchi"));   //in pos. 1

for (Employee e: dipendenti)
        System.out.print(e.getFirstName()+"\n");

int i= dipendenti.size();
dipendenti.get(i); //errore in fase di esecuzione
Employee e1= dipendenti.get(1); //lettura
dipendenti.set(2,e1);                  //scrittura
dipendenti.remove(1);                  //rimozione
```

Gestione identica agli ArrayList, ma tempi differenti!

Iteratori in Java

- permettono di accedere agli elementi dei contenitori
- sono simili agli iteratori del C++
- ma hanno una diversa interfaccia

```
LinkedList<String> s= new
   LinkedList<String>();
s.addLast("Ciao");
...                    //altri inserimenti
Iterator<String> iterator =
   s.iterator();

while(iterator.hasNext())
   System.out.print(iterator.next());
```

Iteratori: operazioni fondamentali

- **`E next()`**

  restituisce l'elemento (della classe E) a cui l'iteratore fa riferimento e passa al successivo elemento

- **`boolean hasNext()`**

  restituisce **true** se ci sono altri elementi e **false** altrimenti

- **`void remove()`**

  elimina l'ultimo elemento ritornato da next(). NON può essere richiamato più volte di seguito dopo una singola chiamata a next().

- Esistono iteratori specifici per i diversi contenitori con metodi dedicati. (es., per **`ListIterator`** esiste il metodo **`previous()`**)

Altri contenitori

In Java esistono altri contenitori. I più importanti sono:

- **HashSet** : insiemi di elementi memorizzati tramite tabelle di hash
- **TreeSet** : insiemi di elementi memorizzti tramite alberi binari di ricerca
- **HashMap** : insieme di coppie <chiave,valore> ordiante con tabelle di hash riepetto alle chiavi
- **TreeMap** : insieme di coppie <chiave, valore> ordinate tramite alberi binari di ricerca rispetto alle chiavi