

## Accesso alle strutture dinamiche usando gli iteratori

Le strutture dinamiche di dati richiedono costantemente l'esecuzione di cicli per scorrere, uno ad uno, tutti gli elementi dell'insieme.

Per eseguire queste iterazioni si può utilizzare la struttura di controllo *for*. Per esempio, un vettore *poligono* contenente oggetti di classe *Punto* può essere attraversato usando il seguente frammento di codice.

```
for(int i=0; i<poligono.size(); i++)
{
    p = (Punto) poligono.elementAt(i);

    // elaborazioni sull'elemento p
}
```

Java consente di gestire in modo generico i cicli sulle strutture di dati dinamiche. Il meccanismo di gestione delle iterazioni è implementato tramite un oggetto di classe **Iterator**, che può essere ricavato da ogni struttura di dati dinamica tramite il relativo metodo **iterator**.

Per esempio, l'oggetto di classe *Iterator* del vettore *poligono* viene dichiarato nel seguente modo:

```
Iterator i = poligono.iterator();
```

Gli oggetti *Iterator* hanno due metodi principali:

- **next()**, restituisce l'elemento successivo dell'insieme;
- **hasNext()**, restituisce *true* se l'*Iterator* contiene ulteriori elementi.

Il ciclo che scorre tutti gli elementi contenuti nell'*Iterator* è il seguente:

```
while (i.hasNext())
{
    p = (Punto) i.next();

    // elaborazioni sull'elemento p
}
```

L'iterazione precedente, sebbene sia simile a quella implementata con il *for*, ha il vantaggio di essere generica, nel senso che può essere utilizzata con diverse strutture dinamiche e non solo con la classe *Vector*.

Consideriamo per esempio la struttura dinamica **ArrayList**, che è una classe contenuta nel package **java.util** ed è molto simile alla classe *Vector*. Gli oggetti di classe *ArrayList* permettono di memorizzare un insieme variabile di elementi (*array*) e vengono creati usando il costruttore vuoto oppure un costruttore che specifica la capacità iniziale.

```
ArrayList elenco1 = new ArrayList();
ArrayList elenco2 = new ArrayList(10);
```

I metodi principali della classe *ArrayList* che consentono di manipolare l'array sono i seguenti:

- **add(Object obj)**, aggiunge un elemento alla fine della lista,
- **remove(int index)**, toglie dal vettore l'elemento nella posizione indicata dall'indice,
- **size()**, restituisce il numero di oggetti effettivamente memorizzati nell'array,
- **get(int index)**, restituisce un oggetto di classe *Object* che si trova nella posizione indicata dall'indice,
- **iterator()**, restituisce un oggetto di classe *Iterator*.

Se costruiamo l'oggetto *poligono* con la classe *ArrayList*,

```
ArrayList poligono = new ArrayList();
```

il ciclo per scorrere i suoi elementi può essere implementato con la struttura *for* nel seguente modo:

```
for(int i=0; i<poligono.size(); i++)
{
    p = (Punto) poligono.get(i);

    // elaborazioni sull'elemento p
}
```

Si noti che è stato usato il metodo *get* per leggere un elemento dell'array, invece del metodo *elementAt* utilizzato con la classe *Vector*.

Se avessimo utilizzato un iteratore, il ciclo di scansione della classe *ArrayList* sarebbe stato implementato nello stesso modo con cui è stato implementato per la classe *Vector*.

```
while (i.hasNext())
{
    p = (Punto) i.next();

    // elaborazioni sull'elemento p
}
```

## Struttura di dati dinamica per gestire le Hash Table

La **Hash Table** (o *tabella hash*) è una struttura di dati dinamica che mette in relazione una **chiave** con il **valore** associato. Le chiavi sono solitamente identificate con una variabile di tipo *String* mentre il valore è un qualsiasi oggetto. Per esempio, l'elenco degli studenti può essere memorizzato in una *Hash Table* la cui chiave è la matricola, mentre il valore è un oggetto contenente le informazioni sullo studente associato.

Il termine *Hash Table* deriva dall'utilizzo di **funzioni di hash** da parte della struttura di dati. La chiave, tramite la funzione di hash, viene trasformata in un indice, utilizzato per accedere alla tabella in cui sono memorizzati i valori.

In Java, la struttura dati che implementa la tabella hash è descritta con la classe **Hashtable**, inclusa nel package **java.util**.

Per dichiarare una tabella hash si possono utilizzare tre costruttori:

```
Hashtable ht = new Hashtable();
```

crea una tabella vuota, con i valori di default per capacità (11) e un fattore di carico (0.75);

```
Hashtable ht = new Hashtable(5);
```

crea una tabella con una capacità iniziale di 5 elementi e un fattore di carico standard (0.75);

```
Hashtable ht = new Hashtable(10, 0.9);
```

crea una tabella con una capacità iniziale di 10 elementi e un fattore di carico 0.9.

Il fattore di carico è un valore tra zero e uno. Con un valore vicino ad uno, si riducono gli spazi lasciati inutilizzati dalla tabella hash ma aumentano i tempi con cui vengono recuperati i valori.

Le principali operazioni che possono essere eseguite su una *Hash Table* sono indicate dai seguenti metodi:

- **put**(chiave, valore), per aggiungere una chiave e un valore alla tabella;
- **get**(chiave), per recuperare il valore associato ad una specifica chiave;
- **remove**(chiave), per cancellare la chiave e il corrispondente valore dalla tabella;
- **contains**(chiave), per verificare se una chiave è presente nella tabella;
- **size**(), per conoscere il numero di elementi nella tabella.

Per accedere a tutti gli elementi della tabella hash in ordine sequenziale si può utilizzare il metodo **iterator**, applicato all'insieme delle chiavi restituite con il metodo **keySet**.

```
Iterator i = ht.keySet().iterator();
```

Il vantaggio di usare una *Hash Table* per memorizzare un insieme di elementi, rispetto ai *Vector* o agli *ArrayList*, è la possibilità di riferirsi direttamente ai singoli elementi tramite una chiave e non tramite un indice numerico.

**Costruire un programma per gestire una rubrica telefonica.**

La rubrica che si vuole gestire contiene un numero variabile di elementi; ogni nominativo della rubrica memorizza un nome e un numero di telefono.

La rubrica viene implementata con una *Hash Table* in cui la *chiave* è il nome, mentre il *valore* è il numero di telefono. Entrambi sono degli oggetti di classe *String*.

Le operazioni che si vogliono gestire sono: l'aggiunta e l'eliminazione di un nominativo dalla rubrica, la visualizzazione della rubrica e la ricerca di un nominativo.

Il programma e l'implementazione della classe che gestisce il menu sono descritti di seguito.

**IMPLEMENTAZIONE DELLA CLASSE** (*Menu.java*)

```
import java.io.*;

class Menu
{
    private InputStreamReader input = new InputStreamReader(System.in);
    private BufferedReader tastiera = new BufferedReader(input);

    private void mostraMenu()
    {
        System.out.println();
        System.out.println("1) Aggiungi nominativo");
        System.out.println("2) Elimina nominativo");
        System.out.println("3) Visualizza rubrica");
        System.out.println("4) Cerca nominativo");
        System.out.println("5) Esci");
    }

    public int scelta()
    {
        int scelta;

        mostraMenu();
        System.out.print("\n-> ");
        try
        {
            String numeroLetto = tastiera.readLine();
            scelta = Integer.valueOf(numeroLetto).intValue();
        }
        catch(Exception e)
        {
            scelta = 0;
        }

        return scelta;
    }
}
```

```

public String leggiDato(String desc)
{
    String dato;

    System.out.print(desc);
    try
    {
        dato = tastiera.readLine();
    }
    catch(Exception e)
    {
        dato = "";
    }

    return dato;
}
}

```

#### PROGRAMMA JAVA (*ProgRub.java*)

```

import java.util.*;

class ProgRub
{
    public static void main(String argv[])
    {
        Hashtable rubrica = new Hashtable(20);
        String chiave, valore;

        Menu mioMenu = new Menu();
        int scelta;

        scelta = mioMenu.scelta();

        while (scelta != 5)
        {
            if (scelta == 1)
            {
                chiave = mioMenu.leggiDato("Nome:");
                valore = mioMenu.leggiDato("Valore:");
                rubrica.put(chiave, valore);
            }
            else if (scelta == 2)
            {
                chiave = mioMenu.leggiDato("Nome:");
                rubrica.remove(chiave);
            }
        }
    }
}

```

```

else if (scelta == 3)
{
    Iterator i = rubrica.keySet().iterator();
    while (i.hasNext())
    {
        chiave = (String) i.next();
        valore = (String) rubrica.get(chiave);
        System.out.println(chiave + " tel. " + valore);
    }
}
else if (scelta == 4)
{
    chiave = mioMenu.leggiDato("Nome:");
    if (rubrica.contains(chiave))
    {
        valore = (String) rubrica.get(chiave);
        System.out.println("Telefono: " + valore);
    }
    else
    {
        System.out.println("Nominativo inesistente.");
    }
}
scelta = mioMenu.scelta();
}

System.out.println("Fine programma.");
}
}

```