Azzolini Riccardo 2018-11-09

Classi generiche `Sequenza<E>` e `SequenzaOrdinata<E>`

1 Classe `Sequenza<E>` e tipi generici

La dimensione degli array deve essere definita a priori. Per memorizzare sequenze di lunghezza sconosciuta, sono quindi necessarie altre strutture dati, come ad esempio la **classe generica** `Sequenza<E>`, le cui istanze rappresentano sequenze di oggetti di un tipo `E`, in ordine di inserimento.

Le classi e i tipi generici sono indicati nella documentazione con una notazione del tipo `Sequenza<E>`.

- `E` viene detto **tipo parametro**.
- Si può dire che la classe (il tipo) è "`Sequenza` di `E`".

La creazione di un oggetto della classe generica `Sequenza` si effettua tramite l'espressione

```
new Sequenza<String>();
// oppure
new Sequenza<Frazione>();
// oppure
new Sequenza<Integer>();
// ecc.
```

che restituisce un riferimento a un oggetto in grado di memorizzare sequenze di oggetti del **tipo argomento** specificato (`String`, `Frazione`, `Integer`, ecc.). Tale tipo può essere un *tipo riferimento* qualunque: per i tipi primitivi, è quindi necessario utilizzare le classi involucro.

I tipi di queste espressioni, cioè `Sequenza<String>`, `Sequenza<Frazione>`, ecc. sono chiamati **tipi parametrizzati** (mentre `Sequenza<E>` è il **tipo generico** e `Sequenza` viene a volte chiamato *raw type*). I riferimenti restituiti dalle espressioni di creazione possono essere memorizzati in variabili dei tipi parametrizzati corrispondenti:

```
Sequenza<String> s = new Sequenza<String>();
// oppure
Sequenza<Frazione> s = new Sequenza<Frazione>();
```

```
// oppure, dalla versione 7 di Java:
Sequenza<Integer> s = new Sequenza<>();
```

1.1 Principali metodi

```
public boolean add(E o)
```

Aggiunge l'oggetto passato come argomento alla fine della sequenza e restituisce `true`. Se invece viene fornito come argomento `null`, non modifica la sequenza e restituisce `false`.

```
public int size()
```

Restituisce il numero di elementi contenuti nella sequenza.

```
public boolean isEmpty()
```

Restituisce `true` se e solo se la sequenza è vuota.

```
public boolean contains(E o)
```

Restituisce `true` se e solo se la sequenza contiene un oggetto uguale all'argomento (in base al metodo `equals` definito per gli oggetti di tipo `E`).

```
public E find(E o)
```

Restituisce un riferimento al primo oggetto nella sequenza uguale all'argomento, oppure `null` se tale oggetto non è presente.

```
public boolean remove(E o)
```

Elimina dalla sequenza il primo elemento uguale all'argomento, se esiste. Restituisce `true` se e solo se è stato effettivamente trovato ed eliminato un elemento.

1.2 Scorrimento degli elementi

È possibile scorrere tutti gli elementi di una sequenza, dal primo all'ultimo, mediante un ciclo for-each:

```
Sequenza<E> sequenza;
// ...
for (E elemento : sequenza) {
    // ... uso di elemento ...
}
```

1.3 Esempio: `PappagalloConMemoria`

```
Sequenza<String> memoria = new Sequenza<String>();

String s = in.readLine();
while (!s.equals("")) {
    memoria.add(s);
    s = in.readLine();
}

for (String x : memoria)
    out.println(x);
```

2 Classe `SequenzaOrdinata<E>`

Le sue istanze rappresentano sequenze ordinate di oggetti di tipo `E`.

I suoi principali metodi hanno prototipi uguali a quelli di `Sequenza<E>`, ma il metodo `add` ha un contratto diverso: esso inserisce il nuovo oggetto oggetto in modo da mantenere ordinata la sequenza (invece che sempre in coda).

Per istanziare il tipo parametro `E`, è necessario un tipo "ordinabile" (a differenza di `Sequenza<E>`, il cui tipo parametro può essere istanziato con qualunque tipo riferimento).

Criteri di ordinamento di alcuni tipi

Tipo	Ordine
`String`	*lessicografico* (alfabetico)
`Integer`	crescente
`Frazione`	crescente
`Data`	cronologico

2.1 Esempio: `PappagalloOrdinato`

```
SequenzaOrdinata<String> memoria = new SequenzaOrdinata<String>();

String s = in.readLine();
while (!s.equals("")) {
    memoria.add(s);
    s = in.readLine();
}
```

```
for (String x : memoria)
    out.println(x);
```

Il codice di questo esempio è identico a quello di `PappagalloConMemoria` (tranne la definizione della variabile `memoria`), ma le stringhe lette vengono stampate in ordine alfabetico.