Azzolini Riccardo 2018-10-16

Cicli ed espressioni

1 Istruzione `while`

```
while (condizione)
    istruzione;
```

Semantica operazionale:

1. Viene valutata la `condizione`
   - se è `true`, viene eseguito il corpo del ciclo e l'esecuzione prosegue dal punto 1
   - se è `false`, l'esecuzione prosegue dall'istruzione successiva

Il corpo può non essere mai eseguito, se la condizione è immediatamente `false`.

2 Istruzione `for`

```
for (espr_inizializzazione; condizione; espr_increment)
    istruzione;
```

- `espr_inizializzazione` è una lista di espressioni, separate da virgola
- `condizione` è un'espressione booleana
- `espr_incremento` è una lista di espressioni
- `istruzione` è un'istruzione singola o un blocco

Semantica operazionale:

1. Vengono valutate le espressioni che compaiono in `espr_inizializzazione`
2. Viene valutata la `condizione`
   - se è `true`
     a) viene eseguito il corpo del ciclo
     b) vengono valutate le espressioni che compaiono in `espr_incremento`
     c) l'esecuzione prosegue dal punto 2
   - se è `false`, l'esecuzione prosegue dall'istruzione successiva al ciclo `for`

Varianti:

- `espr_inizializzazione`, `condizione` e `espr_incremento` possono anche essere vuote.
- `espr_inizializzazione` può contenere direttamente la dichiarazione della variabile di controllo
  - in questo caso, la variabile non è definita al di fuori del ciclo
  - è possibile dichiarare più variabili, purché siano tutte dello stesso tipo

2.1 Esempi

```
int cont;
for (cont = 1; cont <= 10; cont = cont + 1)
    out.println(cont);

for (; x != 0; ) { // equivalente a while (x != 0)
    x = in.readInt("Inserisci un numero ");
    out.println(x);
}

for (;;) // ciclo infinito, equivalente a while (true)
    ...

for (int cont = 1; cont <= 10; cont = cont + 1)
    out.println(cont);

for (int i = 1, j = 0; i + j <= 20; i = i + 1, j = j + 1)
    out.println(i + j);
```

3 Istruzione `break`

Interrompe l'esecuzione del corpo del ciclo in cui compare e termina l'iterazione.

3.1 Esempio

Verifica se una stringa è palindroma:

```java
String s = "radar";
boolean palindroma = true;

for (int sx = 0, dx = s.length() - 1; sx < dx; sx = sx + 1, dx = dx + 1)
    if (s.charAt(sx) != s.charAt(dx)) {
        palindroma = false;
        break;
    }
```

4 Istruzione `continue`

Provoca l'interruzione dell'esecuzione corrente del corpo del ciclo e il passaggio all'iterazione successiva:

- nei cicli `while` e `do...while`, si passa immediatamente alla valutazione della condizione (e, in base ad essa, all'iterazione successiva o all'uscita dal ciclo)
- nel ciclo `for`, vengono eseguite le espressioni di incremento, quindi viene valutata la condizione

5 Espressione

Un'**espressione** è una porzione di codice Java caratterizzata da

- un **tipo** determinato al momento della *compilazione*
- un **valore** determinato al momento dell'*esecuzione*

5.1 Espressioni di base

- **letterali** (ad esempio, `25` è un letterale di tipo `int`, mentre `"pippo"` è un letterale di tipo `String`)
- **variabili**
  - *tipo*: il tipo associato alla variabile in fase di dichiarazione
  - *valore*: il valore contenuto nella variabile
- **invocazioni di metodo**
  - *tipo*: il tipo restituito dal metodo
  - *valore*: il risultato dell'esecuzione del metodo

5.2 Espressioni composte

Sono costituite da espressioni di base, combinate tra con degli **operatori**.

6 Operatori binari di tipo `int`

I principali sono:

- `+`
- `-`
- `*`
- `/` (divisione intera)
- `%` (resto della divisione)

Hanno le stesse regole di precedenza definite in algebra, ed è possibile modificarle utilizzando le parentesi tonde.

`+` e `-` fungono anche da operatori unari prefissi (con precedenza superiore a tutti gli operatori binari).

7 Operatore di assegnamento

L'operatore `=` dà luogo a espressioni del tipo `variabile = espressione`

- *tipo*: il tipo della `variabile` a sinistra dell'operatore
- *valore*: il valore dell'`espressione` a destra dell'operatore

Il tipo dell'`espressione` deve essere **compatibile** con quello della `variabile`, cioè il valore dell'`espressione` deve essere **assegnabile** alla `variabile`.

Un'espressione di assegnamento seguita dal punto e virgola costituisce un'**istruzione di assegnamento**.

7.1 Esempi

- Se `x` è una variabile di tipo `int`,

  `x = 10`

  è un'espressione corretta con

  - tipo: `int`
  - valore: 10

- Se `s` è una variabile di tipo `String`,

  `s = "pippo".toUpperCase()`

  è un'espressione corretta con

  - tipo: `String`
  - valore: riferimento a un oggetto che rappresenta la stringa `"PIPPO"`

- Se `x` e `y` sono variabili di tipo `int`,

  `x = y = 10`

  è un'espressione corretta con

  - tipo: `int`
  - valore: 10

  perché viene valutata come:

  `x = (y = 10)`