Informatik 11: Einführung in Java — Kara kann auch zählen!

Gierhardt

Aufgabe: KARA soll vor sich eine Spur von fünf Kleeblättern legen.

Das Neue: KARA merkt sich die Anzahl der schon abgelegten Kleeblätter in einem internen „Gedächtnis“, d.h. konkret in einem Speicher. Die Speicherplätze sind durch Variablen ansprechbar.

1. **Lösung mit einer While-Schleife:**

```
import JavaKaraProgram;
public class Zaehlen extends JavaKaraProgram
{//Anfang von Zaehlen
  public void myProgram()
  {  // Anfang von myProgram
  int i;
  i = 0;
  while (i<5)
    {
      kara.putLeaf();
      kara.move();
      i = i+1;
    }
  }  // Ende von myProgram
} // Ende von Zaehlen
```

Erläuterungen:

(a) Mit `int i;` wird Speicherplatz für eine Variable mit dem Namen `i` und dem Typ `int`, d.h. INTEGER, reserviert. Man sagt: Die Variable `i` wird *deklariert*. Java kennt fünf verschiedene INTEGER-Typen:

Typ	von	bis einschließlich	Größe
`byte`	-128	127	8 Bit
`short`	-32.768	32.767	16 Bit
`int`	-2.147.483.648	2.147.483.647	32 Bit
`long`	-9.223.372.036.854.775.808	9.223.372.036.854.775.807	64 Bit
`char`	0	65535	16 Bit

(b) Durch `i = 0` wird der Variablen `i` der Wert 0 zugewiesen. Man sagt: Die Variable `i` wird *initialisiert.*

(c) Deklaration und Initialisierung können zusammengefasst werden durch

```
int i = 0;
```

(d) Für die Bedingung `i<5` wird der Vergleichsoperator `<` benutzt. Java kennt die folgenden Vergleichsoperatoren:

Zeichen	Bedeutung
`<`	kleiner als
`>`	größer als
`<=`	kleiner oder gleich
`>=`	größer oder gleich
`==`	gleich
`!=`	ungleich
`instanceof`	Type-Vergleich (kommt später dran!)

Zur Erinnerung: „Mich hat man ja nicht gefragt bei der Entwicklung von Java!“

(e) Die Zuweisung `i = i + 1;` muss man von rechts nach links lesen: „Nimm den aktuellen Wert von `i`, addiere 1 dazu und speichere den neuen Wert wieder unter dem Namen `i` ab.

Für eine Addition von 1 gibt es eine abkürzende Schreibweise: `i++;`

(f) Es ist möglich, eine Variable mit einem endgültigen, unveränderlichen Wert zu versehen, d.h. sie zu einer Konstanten zu machen:

```
final int ANZAHL=5;
```

Dann könnte man im obigen Programmbeispiel `while (i<ANZAHL)` schreiben.

Konstanten schreibt man komplett mit Großbuchstaben.

(g) Variablen schreibt man wie Methoden beginnend mit einem Kleinbuchstaben.

(h) In einer Zeile lassen sich mehrere Variablen zugleich deklarieren bzw. initialisieren:

```
int x, y, z, a=1, zahl;
```

2. **Lösung mit einer for-Schleife:**

Variante 1:

```
import JavaKaraProgram;
public class Zaehlen extends JavaKaraProgram
{  // Anfang von Zaehlen
  public void myProgram()
  {  // Anfang von myProgram
  int i;
  for (i=1; i<=5; i=i+1)
    {
      kara.putLeaf();
      kara.move();
    }
  }  // Ende von myProgram
} // Ende von Zaehlen
```

Variante 2:

```
import JavaKaraProgram;
public class Zaehlen extends JavaKaraProgram
{  // Anfang von Zaehlen
  public void myProgram()
  {  // Anfang von myProgram
  for (int i=1; i<=5; i++)
    {
      kara.putLeaf();
      kara.move();
    }
  }  // Ende von myProgram
} // Ende von Zaehlen
```

Erläuterungen:

(a) Die Syntax der `for`-Schleife:

```
for ( Startwert; Bedingung; Zaehlweise )
  {
    Anweisungen
  }
```

(b) Ablauf: Zuerst wird die Schleifenvariable mit dem Startwert initialisiert und dann die Bedingung geprüft. Ist sie wahr, werden die Anweisungen ausgeführt. Am Ende der Schleife wird die Schleifenvariable nach Angabe der Zählweise verändert. Danach werden nur noch zu Beginn der Schleife die Bedingung überprüft, die Anweisungen ausgeführt und jeweils am Ende die Schleifenvariable verändert, bis die Bedingung falsch wird.

(c) Bei Variante 2 wird die Schleifenvariable `i` sogar innerhalb der Klammer von `for` deklariert und initialisiert. Dies hat zur Folge, das die Variable `i` nur innerhalb der Schleife Gültigkeit hat. Man sollte sich angewöhnen, nur nach Variante 2 zu arbeiten!

(d) Achtung! Kleiner Fehler mit großer Wirkung!

```
for (int i=1; i<=5; i++);
  {
    kara.putLeaf();
    kara.move();
  }
```