Tutorium 5: Beispielprogramme

17. November 2009

1 Wozu Methoden?

- Wiederverwendbarkeit
- Strukturierung des Programmcodes

Das folgende Programm verwendet Anweisungen für den Refrain mehrfach.

```
1  /* Diese Klasse singt Knockin' on Heaven's Door auf der Konsole.
2   */
3  class HeavensDoor {
4
5      public static void main(String[] args) {
6
7          System.out.println();
8
9          // Strophe 1
10         System.out.println("Mama, take this badge off of me");
11         System.out.println("I can't use it anymore.");
12         System.out.println("It's gettin' dark, too dark for me to see");
13         System.out.println("I feel like I'm knockin' on heaven's door.");
14
15         // Refrain
16         System.out.println();
17         for(int i = 0; i < 4; i++) {
18             System.out.println("Knock, knock, knockin' on heaven's door");
19         }
20         System.out.println();
21
22         // Strophe 2
23         System.out.println("Mama, put my guns in the ground");
24         System.out.println("I can't shoot them anymore.");
25         System.out.println("That long black cloud is comin' down");
```

```
26          System.out.println("I feel like I'm knockin' on heaven's door.");
27
28          // Refrain
29          System.out.println();
30          for(int i = 0; i < 4; i++) {
31              System.out.println("Knock, knock, knockin' on heaven's door");
32          }
33          System.out.println();
34      }
35  }
```

2 Methoden ohne Rückgabewert und ohne Parameter

Das folgende Programm gibt den Refrain einen Namen (`singeRefrain()`). Durch Aufruf des Namens werden die Anweisungen des Refrains ausgeführt. Programm wird dadurch strukturiert und der Code des Refrains kann auf einfache Weise wiederverwendet werden. Eventuelle Fehler im Refrain brauchen nur einmal in der Methode `singeRefrain()` korrigiert werden statt mehrfach wie in der ersten Version des Programms.

```
1   /* Diese Klasse singt Knockin' on Heaven's Door auf der Konsole.
2    *
3    * Version 2
4    */
5   class HeavensDoor2 {
6
7           /* Diese Methode singt den Refrain.
8            */
9           public static void refrainSingen() // Methodenkopf
10          { // Methodenblock, Rumpf, Implementation
11              System.out.println();
12              for(int i = 0; i < 4; i++) {
13                  System.out.println("Knock, knock, knockin' on heaven's door");
14              }
15              System.out.println();
16          }
17
18          public static void main(String[] args) {
19              System.out.println();
20
21              // Strophe 1
22              System.out.println("Mama, take this badge off of me");
23              System.out.println("I can't use it anymore.");
24              System.out.println("It's gettin' dark, too dark for me to see");
```

```
25              System.out.println("I feel like I'm knockin' on heaven's door.");
26
27              // Refrain
28              refrainSingen();
29
30              // Strophe 2
31              System.out.println("Mama, put my guns in the ground");
32              System.out.println("I can't shoot them anymore.");
33              System.out.println("That long black cloud is comin' down");
34              System.out.println("I feel like I'm knockin' on heaven's door.");
35
36              // Refrain
37              refrainSingen();
38          }
39  }
```

3 Methoden ohne Rückgabewert und mit Parameter

```
1   /* Diese Klasse singt Knockin' on Heaven's Door auf der Konsole.
2    *
3    * Version 3
4    */
5   class HeavensDoor3 {
6       /* Diese Methode singt den Refrain. Der übergebene Parameter gibt an,
7        * wie oft der Refrain wiederholt wird.
8        *
9        * Parameter:
10       * int n - Anzahl der Wiederholungen
11       */
12      public static void refrainSingen(int n) {
13          System.out.println();
14          for(int i = 0; i < n; i++) {
15              System.out.println("Knock, knock, knockin' on heaven's door");
16          }
17          System.out.println();
18      }
19
20      public static void main(String[] args) {
21
22          System.out.println();
23
24          // Strophe 1
```

```
25          System.out.println("Mama, take this badge off of me");
26          System.out.println("I can't use it anymore.");
27          System.out.println("It's gettin' dark, too dark for me to see");
28          System.out.println("I feel like I'm knockin' on heaven's door.");
29
30          // Refrain
31          refrainSingen(2);
32
33          // Strophe 2
34          System.out.println("Mama, put my guns in the ground");
35          System.out.println("I can't shoot them anymore.");
36          System.out.println("That long black cloud is comin' down");
37          System.out.println("I feel like I'm knockin' on heaven's door.");
38
39          // Refrain
40          refrainSingen(4);
41      }
42  }
```

4 Methoden mit Rückgabewert und mit Parameter

Beachtet auch die unterschiedlichen Arten mit der die Methode `y(x)` in der `main()`-Methode aufgerufen werden kann.

```
1   /* Diese Klasse implementiert
2    */
3   class Polynom {
4
5       /* Diese Methode berechnet das Polynom
6        * y = x^2 + 1
7        *
8        * Parameter:
9        * double x - Wert
10       * Rückgabewert:
11       * double - x^2 +1
12       */
13      public static double y(double x) {
14          double result = x*x + 1.0;
15          // letze Anweisung der Methode
16          return result;
17      }
18
19      public static void main(String[] args) {
```

```
20
21          // Methodenaufruf Variante 1
22          double val1 = y(2.0);
23          System.out.println("y(" + 2.0 + ") = " + val1);
24
25          // Methodenaufruf Variante 2
26          double x2 = 2.0;
27          double val2 = y(x2);
28          System.out.println("y(" + x2 + ") = " + val2);
29
30          // Methodenaufruf Variante 3
31          double x3 = 4.0;
32          double val3 = y((x3 + 2.0)/3.0);
33          System.out.println("y(" + ((x3 + 2.0)/3.0) + ") = " + val3);
34
35          // Methodenaufruf Variante 4
36          double x4 = 2.0;
37          System.out.println("y(" + x4 + ") = " + y(x4));
38      }
39  }
```

```
1   /* Diese Klasse implementiert einige mathematische Methoden.
2    */
3   class MyMath {
4
5       /* Diese Methode liefert den Preis inklusive Mehrwertsteuer zurueck.
6        * Parameter:
7        * double wert - Preis ohne Mwst
8        * double mwst - Mehrwertsteuer
9        * Rueckgabewert:
10       * double - Preis inklusive Mwst
11       *
12       */
13      public static double berechnePreis(double wert, double mwst) {
14          double p = wert + wert*mwst;
15          return p;
16      }
17      // Alternative
18      public static double berechnePreis2(double wert, double mwst) {
19          return wert + wert*mwst;
20      }
21
22      /* Diese Methode berechnet den Betrag des übergebenen Parameters
23       * Parameter:
```

```
24      * double x - Wert
25      * Rueckgabewert:
26      * double - Betrag |x|
27      */
28     public static double abs(double x) {
29         if(x < 0) {
30             return -x;
31         } else {
32             return x;
33         }
34     }
35
36     // Alternative zur Methode abs(x)
37     public static double abs2(double x) {
38         if(x < 0) {
39             return -x;
40         }
41         return x;
42     }
43
44     /* Diese Methode liefert den Wert true zurueck genau dann, wenn der ueber-
45      * gebene Array die Zahl Null enthaelt.
46      * Parameter:
47      * int[] arr - Array von Ganzzahlen (mindestens ein Element)
48      * Rueckgabewert:
49      * boolean - true gdw 0 Element von arr
50      */
51     public static boolean hasZero(int[] arr) {
52         for(int i = 0; i < arr.length; i++) {
53             if(arr[i] == 0) {
54             return true;
55             }
56         }
57         return false;
58     }
59
60     // Test
61     public static void main(String[] args) {
62
63         // Preis inkl. Mwst
64         //System.out.println("Preis (+ Mwst) = " + berechnePreis(100.0, 0.19));
65
66         // Absolutbetrag
67         System.out.println("Betrag von -2.0 = " + abs(-2));
```

```
68          System.out.println("Betrag von 0.0 = " + abs(0));
69          System.out.println("Betrag von +2.0 = " + abs(2));
70
71          // hasZero
72          int[] arr1 = {1, 4, 8, 2, 0, 9};
73          System.out.println("arr1 enthaelt eine 0 = " + hasZero(arr1));
74          int[] arr2 = {2, 5, 7, 2, 1, 3, 8, 8};
75          System.out.println("arr2 enthaelt eine 0 = " + hasZero(arr2));
76      }
77  }
```

5 Rekursionen

```
1   /* Diese Klasse implementert die Fakultaetsfunktion rekursiv und iterativ.
2    */
3   class Fakultaet {
4
5       /* Diese Methode berechnet rekursiv die Fakultaet des uebergebenen
6        * Parameters.
7        * Parameter:
8        * int n - eine natuerliche Zahl
9        * Rueckgabewert:
10       * int - Fakultaet von n
11       */
12      public static int fak(int n) {
13          if(n <= 1) {
14              // Rekursionsanker
15              return 1;
16          } else {
17              // rekursiver Aufruf
18              return n*fak(n-1);
19          }
20      }
21
22      /* Diese Methode berechnet iterativ die Fakultaet des uebergebenen
23       * Parameters.
24       * Parameter:
25       * int n - eine natuerliche Zahl
26       * Rueckgabewert:
27       * int - Fakultaet von n
28       */
29      public static int fakIter(int n) {
```

```
30          int val = 1;
31          for(int i = 1; i <= n; i++) {
32              val = val * i;
33          }
34          return val;
35      }
36
37
38      public static void main(String[] args) {
39          int n = Integer.parseInt(args[0]);
40          System.out.println("Rekursion: n! = " + fak(n));
41          System.out.println("Iteration: n! = " + fakIter(n));
42      }
43  }
```

```
1   /* Diese Klasse implementert die folgende Zahlenfolge rekursiv und iterativ:
2    *
3    * f(1) = 2
4    * f(n) = 2*f(n-1) + 1 fuer n < 1
5    */
6   class Rekursion {
7
8       /* Diese Methode berechnet rekursiv den oben spezifizierten Wert f(n),
9        * wobei n als Parameter uebergeben wird.
10       *
11       * Parameter:
12       * int n - eine natuerliche Zahl
13       * Rueckgabewert:
14       * int - f(n)
15       */
16      public static int f(int n) {
17          if(n <= 1) {
18              // Rekursionsanker
19              return 2;
20          } else {
21              // rekursiver Aufruf
22              return 2*f(n-1) + 1;
23          }
24      }
25
26      /* Diese Methode berechnet iterativ den oben spezifizierten Wert f(n),
27       * wobei n als Parameter uebergeben wird.
28       *
29       * Parameter:
```

```
30        * int n - eine natuerliche Zahl
31        * Rueckgabewert:
32        * int - f(n)
33        */
34       public static int fIter(int n) {
35           int val = 2;
36           for(int i = 2; i <= n; i++) {
37               val = 2*val + 1;
38           }
39           return val;
40       }
41
42       // Test
43       public static void main(String[] args) {
44           int n = Integer.parseInt(args[0]);
45           System.out.println("Rekursion: f(" + n + ") = " + f(n));
46           System.out.println("Iteration: f(" + n + ") = " + fIter(n));
47       }
48   }
```