Informatik: Einführung in Java Rekursion mit Parametern (Lsg.)

Gierhardt

Aufgaben:

1. Schreibe eine Methode `potenz`, die die n-te Potenz einer Zahl x rekursiv berechnet.

```
1      double potenz(double x, int n)
2      {   if      (n==0) return 1;
3          else if (n==1) return x;
4          else           return x * potenz(x, n-1);
5      }
```

2. Schreibe eine Methode `fakultaet`, die die Fakultät einer natürlichen Zahl berechnet.

```
1    int fakultaet(int n)
2    {   if (n==0)
3            return 1;
4     else   return n * fakultaet(n-1);
5    }
```

3. Schreibe eine Methode `fibo`, die die n-te Fibonacci-Zahl berechnet.

```
1    int fibo(int n)
2    {   if (n<=2)
3              return 1;
4        else return fibo(n-1) + fibo(n-2);
5    }
```

4. Kara soll eine Figur aus Kleeblättern nach folgendem Schema legen.

   - In der ersten Zeile liegt ein Kleeblatt.
   - In der zweiten Zeile liegt ein Kleeblatt.
   - In jeder anderen Zeile liegen so viele Kleeblätter wie in den beiden darüber liegenden Zeilen zusammen.

```
1  import javakara.JavaKaraProgram;
2
3  public class FiboFeld extends JavaKaraProgram
4  {
5    void turnAround()
6    {   kara.turnLeft();
7        kara.turnLeft();
8    }
9
10   void vorX(int n, boolean blatt)
11   {   if (n>0)
```

```
12            { if (blatt) kara.putLeaf();
13              kara.move();
14              vorX(n-1, blatt);
15            }
16    }
17
18    int fibo(int n)
19    {   if (n<=2) return 1;
20        else return fibo(n-1) + fibo(n-2);
21    }
22
23    void fiboFeld(int maxIndex)
24    {   for (int i=1; i<=maxIndex; i++)
25        {       vorX(fibo(i),true);   // Steht hinter letztem Blatt
26                turnAround();         // zurueck
27                vorX(fibo(i),false); // und keine Blaetter legen
28                kara.turnLeft();      // zur maechsten Zeile
29                kara.move();
30                kara.turnLeft();
31        }
32    }  // Ende von fiboFeld
33
34    public void myProgram()
35    { fiboFeld(8);
36    }
37  }  // Ende von FiboFeld
```

5. Schreibe eine rekursive Version der Methode `geheX(int n)`.

   (siehe vorhergehende Aufgabe)

6. Die beiden Methoden `geheX(int n)` und `legeX(int n)` sollen zu einer neuen Methode `vorX(int n, boolean blatt)` zusammengefasst werden. Bei `blatt=true` wird ein Blatt beim Weitergehen gelegt, sonst nicht. Die Methode ist rekursiv zu formulieren.

   (siehe vorhergehende Aufgabe)

7. KARA soll wie dargestellt eine Spirale legen und an seinen Startpunkt zurückkehren. Benutze dazu die vorher formulierte Methode `vorX(int n, boolean blatt)`.

```
1  import javakara.JavaKaraProgram;
2  public class Spirale extends JavaKaraProgram
3  {
4    void turnAround()
5    { kara.turnLeft();
6      kara.turnLeft();
7    }
8
9    void vorX(int n, boolean blatt)
10   {   if (n>0)
11           { if (blatt) { kara.putLeaf(); }
12             kara.move();
13             vorX(n-1, blatt);
14           }
15   }
16
17   void spirale(int laenge)
18   {   if (laenge>=1)
19           {   vorX(laenge,true); // Steht hinter letztem Blatt
20               turnAround();      // Nach rechts
21               kara.move();       // zum naechsten
22               kara.turnLeft();   // freien
23               kara.move();       // Feld.
24               spirale(laenge-1); // Rekursiver Aufruf
25               kara.turnLeft();   // und
26               vorX(laenge,false);// zurueck
27           }
28       else {   turnAround(); // Drehen und
29               kara.move();   // einen Schritt zurueck
30           }
31   }  // Ende von spirale
32
33   public void myProgram()
34   {
35     spirale(12);
36   }
37 }  // Ende von Spirale
```

8. KARA soll in der Mitte einer Spirale beginnen und die Spirale so legen, dass in den „Seitenlängen“ die FIBONACCI-Zahlen auftreten.

```
1  import javakara.JavaKaraProgram;
2  public class FiboSpirale extends JavaKaraProgram
3  {
4    void turnAround()
5    {  kara.turnLeft();
6       kara.turnLeft();
7    }
8
9    void vorX(int n, boolean blatt)
10   {  if (n>0)
11         { if (blatt) { kara.putLeaf(); }
12           kara.move();
13           vorX(n-1, blatt);
14         }
15   }
16
17   int fibo(int n)
18   {  if (n<=2) return 1;
19      else return fibo(n-1) + fibo(n-2);
20   }
21
22   void fiboSpirale(int i, int maxIndex)
23   // zeichnet von innen nach aussen eine Spirale,
24   // deren "Seitenlaengen" den Fibonaccizahlen mit dem
25   // Index 1 bis zum Index maxIndex entsprechen.
26   {  if (i <= maxIndex)
27         {  vorX(fibo(i)-1,true);  // Steht hinter letztem Blatt
28            kara.turnRight();      // Nach rechts
29            fiboSpirale(i+1, maxIndex);   // Rekursiver Aufruf
30         }
31   }  // Ende von fiboSpirale
32
33   public void myProgram()
34   {
35     fiboSpirale(1, 8);
36   }
37 }  // Ende von FiboSpirale
```

9. Was ist der Funktionswert `makkaroni(5)`?

```
int makkaroni(int nudel)
{   if (nudel==1)
          { return 1; }
    else { if (nudel==2)
                 return 2;
           else return 2*makkaroni(nudel-2)+makkaroni(nudel-1);
          }
}
```

nudel	makkaroni(nudel)
1	1
2	2
3	4
4	8
5	16

10. Was ist der Funktionswert `brat(5)`?

```
int brat(int kartoffel)
{   if (kartoffel==1)
          { return 1; }
    else { if (kartoffel==2)
                 return 3;
           else return 3*brat(kartoffel-2)+2*brat(kartoffel-1);
          }
}
```

kartoffel	brat(kartoffel)
1	1
2	3
3	9
4	27
5	81