1. In der in JavaKara eingebauten Methodensammlung „fehlen“ einige Methoden. Es gibt `treeFront()`, `treeLeft()` und `treeRight()` für Bäume, weil KARA dafür Sensoren besitzt. Für Kleeblätter gibt es nur den Sensor `onLeaf()`. Wir können aber einen Sensor für Kleeblätter um KARA herum nachbilden, in dem wir KARA dort hinschicken, mit `onLeaf()` nachsehen und zurück kehren lassen. Das Ergebnis seiner Erkundung (`true` oder `false`) soll KARA zurück liefern.

   Schreibe den Java-Code für `leafLeft()`, `leafRight()` und `leafFront()`. Findet KARA an der entsprechenden Stelle einen Baum, so soll `false` zurück geliefert werden.

   (Lösung: siehe Aufgabe 2)

2. KARA steht vor einem ihm fremden Wald, denn darin befinden sich keinerlei Kleeblätter. Er erinnert sich an das Märchen von Hänsel und Gretel und deren Strategie, im fremden Wald Brotkrümel zu streuen. Statt der Brotkrümel legt er beim Hineingehen in den Wald Kleeblätter ab, um den Rückweg wieder finden zu können. Er startet am Waldrand an einem Pilz, geht hinein und verirrt sich tatsächlich im Wald. In dieser Situation befindet er sich nun beim Start des zu schreibenden Programmes. Er soll also seiner gelegten Kleeblattspur entlang zurück zum Pilz laufen. Es empfiehlt sich, die neu eingerichteten „Sensoren“ von Aufgabe 1 zu benutzen.

```
1  import javakara.JavaKaraProgram;
2
3  public class HaenselUndGretel extends JavaKaraProgram
4  {
5    void turnAround()
6    { kara.turnLeft();
7      kara.turnLeft();
8    }
9
10   void moveBack()
11   { turnAround();
12     kara.move();
13     turnAround();
14   }
15
16   boolean leafFront()
17   { boolean blatt=false; // damit false fuer Baum
18     if (!kara.treeFront())
19           { kara.move();
20             blatt=kara.onLeaf();
21             moveBack();
22           }
23     return blatt;
24   }
25
```

```
26    boolean leafLeft()
27    { boolean blatt=false; // damit false fuer Baum
28      if (!kara.treeLeft())
29           { kara.turnLeft();
30             kara.move();
31             blatt=kara.onLeaf();
32             moveBack();
33             kara.turnRight();
34           }
35      return blatt;
36    }
37
38    public void myProgram()
39    { while ( !kara.mushroomFront() )
40        { if (!leafFront())
41                { if (leafLeft()) { kara.turnLeft(); }
42                  else { kara.turnRight(); }
43                }
44          kara.move();
45        }
46      tools.showMessage("Ich bin draussen!");
47    }  // Ende von myProgram
48  }  // Ende von HaenselUndGretel
```

3. Das vorherige Programm soll so modifiziert werden, dass er irgendwo am Waldrand gestartet war, sich aber die Zahl 20 für die Anzahl der abgelegten Kleeblätter bis zum Verirren gemerkt hat. Beim Ablegen des siebten Kleeblattes hat er sich noch gemerkt, dass an dieser Stelle eine sehr schöne Lichtung mit vielen wunderschönen Marienkäferinnen war. Auf dem Rückweg soll er bei der schönen Lichtung einen Gruß ausgeben und dann weiter zurück bis zu seinem Startpunkt am Waldrand gehen.

   Lösung ähnlich wie oben, nur diesmal mit `for`-Schleife.

4. Kara sieht vor sich eine freie Strecke. Links von dieser Strecke gehen ununterbrochen verschieden lange Kleeblattspuren ab. Diese Kleeblattspuren sollen an seiner Laufstrecke entlang nach rechts gespiegelt werden. Benutze zur Lösung des Problems eine Methode `spurlaenge()`, die die Länge der aktuellen Spur links zurück liefert. Die schon programmierten Methoden `geheX()` und `legeX()` dürfen/sollen natürlich auch benutzt werden.

```
1  import javakara.JavaKaraProgram;
2
3  public class Spiegeln extends JavaKaraProgram
4  {
5    void turnAround()
6    { kara.turnLeft();
7      kara.turnLeft();
8    }
9
10   void moveBack()
11   { turnAround();
12     kara.move();
13     turnAround();
14   }
15
16   void legeX(int anzahl)
17   { for (int i=1; i<=anzahl; i++)
18       { kara.putLeaf();
19         kara.move();
20       }
21   }
22
23   void geheX(int anzahl)
24   { for (int i=1; i<=anzahl; i++)
25       { kara.move(); }
26   }
27
28   boolean leafLeft()
29   { boolean blatt=false; // damit false fuer Baum
30     if (!kara.treeLeft())
31           { // jetzt muss man nachschauen
32             kara.turnLeft();
33             kara.move();
34             blatt=kara.onLeaf();
35             moveBack();
36             kara.turnRight();
37           }
38     return blatt;
39   }
40
41   int spurlaenge()
42   {  int zaehler=0;
43      kara.turnLeft();
44      kara.move();
45      while (kara.onLeaf())
46        {  zaehler++;
```

```
47              kara.move();
48            }
49        turnAround();
50        geheX(zaehler+1);
51        kara.turnLeft();
52        return zaehler;
53    }
54
55    public void myProgram()
56    {  // Anfang von myProgram
57      int anzahl=0;
58      while (!leafLeft()) { kara.move(); }
59      while (leafLeft())
60        {  anzahl=spurlaenge();
61           kara.turnRight();
62           kara.move();
63           legeX(anzahl);
64           turnAround();
65           geheX(anzahl+1);
66           kara.turnRight();
67           kara.move();
68        }
69    }  // Ende von myProgram
70  }  // Ende von Spiegeln
```