Übungsblatt 8 (Übungsklausur)

Musterlösung

Aufgabe 1 (50 Punkte)

Es werden zwei Attribute für die Position des Schatzes benötigt. Diese könnten auch *final* sein, aber ab Aufgabe 2 geht das nicht mehr, weil die Werte dort erst berechnet werden.

```
1     private int x;
2     private int y;
```

Im Konstruktor werden die zwei Attribute auf die Werte der Parameter gesetzt:

```
3         this.x = x;
4         this.y = y;
```

In der Methode *bewege* muss dann geschaut werden, in welcher Richtung der Schatz liegt, und die Figur geeignet gedreht werden. Ist sie bereits beim Schatz, wird *false* zurückgegeben, ansonsten *true*.

```
5         if (x > figur.getX()) {
6             figur.setRotation(0);
7         }
8         else if (x < figur.getX()) {
9             figur.setRotation(2);
10        }
11        else if (y > figur.getY()) {
12            figur.setRotation(1);
13        }
14        else if (y < figur.getY()) {
15            figur.setRotation(3);
16        }
17        else {
18            return false;
19        }
20        return true;
```

Aufgabe 2 (25 Punkte)

Im Konstruktor wird das 'X' gefunden, indem das Array und dann die jeweilige Zeile durchsucht wird. Für die Suche in der Zeile könnte alternativ auch die *String*-Methode *indexOf* verwendet werden. An der Fundstelle können die Schleifenvariablen an x und y zugewiesen werden oder es werden einfach direkt x und y als Schleifenvariablen verwendet und die Schleife an der Fundstelle verlassen.

```
1         for (y = 0; y < karte.length; ++y) {
2             for (x = 0; x < karte[y].length(); ++x) {
3                 if (karte[y].charAt(x) == 'X') {
4                     return;
5                 }
6             }
7         }
```

Aufgabe 3 (25 Punkte)

Hier gibt es mindestens zwei Möglichkeiten, die Aufgabe zu lösen. Zum einen kann bei Erreichen des Zieles wieder die Startposition als Ziel gesetzt werden, zum anderen kann der zurückgelegte Weg gespeichert werden. Bei der ersten Lösung muss allerdings für den Rückweg eine andere Reihenfolge bei den Abfragen zur Richtungswahl genutzt werden, da ansonsten nicht derselbe Weg verwendet wird. Bei beiden Varianten wird ein Attribut benötigt, das speichert, ob sich die Figur gerade auf dem Hin- oder Rückweg befindet.

Hier die Variante mit der Speicherung des Wegs, bei der alle Schritte auf dem Hinweg in einer *ArrayList* gespeichert werden und diese dann auf dem Rückweg von hinten wieder abgebaut wird.

Zwei neue Attribute werden benötigt (und ein *import java.util.ArrayList;*):

```
1     private final ArrayList<Integer> schritte = new ArrayList<>();
2     private boolean rückweg = false;
```

In der Methode *bewege* wird vor dem letzten *else* der Code ergänzt, der zwischen Hin- und Rückweg unterscheidet:

```
3         else {
4             rückweg = true;
5         }
6
7         if (!rückweg) {
8             schritte.add(figur.getRotation());
9         }
10        else if (schritte.size() > 0) {
11            figur.setRotation(schritte.removeLast() + 2);
12        }
```