Eine Diamantminenfirma hat vom großen Erfolg des ReaktorRobots gehört und überlegt, ob sie nicht auch die gefährlichen Arbeiten in den Diamantenminen durch Roboter ausführen lassen soll, da es immer wieder zu Stolleneinbrüchen kommt oder gefährliche Sprengungen notwendig sind. Eine letzte Aufgabe für unsere Rescue-Robots … zum Glück kommen diese gerade aus dem Trainingslager. Schau mal an, was sie dort gelernt haben.

Gezählte Wiederholungen

ZIEL: *Zählschleifen kennen und (in mindestens einer Art) in Java notieren können.*

Die Methode rundeDrehen() in der Klasse AB10 wurde umständlich notiert:

```
public void rundeDrehen() {
  laufeBisWand();
  dreheLinks();
  laufeBisWand();
  dreheLinks();
  laufeBisWand();
  dreheLinks();
  laufeBisWand();
  dreheLinks();
}
```

Der Roboter umrundet gegen den Uhrzeigersinn ein von Wänden eingeschlossenes rechteckiges Feld.

Dabei macht er vier Mal das Gleiche. Sehr häufig sind einige Befehle zu wiederholen. Daher gibt es in den Programmiersprachen die Möglichkeit, solche Wiederholungen zu notieren.

Wir müssen dem Roboter mitteilen, was er wiederholen soll [laufeBisWand() und dreheLinks()] und wie oft. Dabei muss er selbst mitzählen, wie oft er es schon gemacht hat.

```
public void rundeDrehen() {
  int anzGemacht;
  anzGemacht = 0;
  while (anzGemacht < 4) {
    laufeBisWand();
    dreheLinks();
    anzGemacht++;
  }
}
```

Er muss also SOLANGE die Befehle erneut ausführen wie die Rundenzahl kleiner ist als 4.

Die Wiederholungsrunden muss er zählen. Dafür benötigt er innerhalb der Methode rundeDrehen() eine Variable. Sie wird als ganzzahlige Variable mit dem Namen anzGemacht für die Wiederholung bereitgestellt: int anzGemacht; Die Anweisung anzGemacht = 0; legt den Anfangswert der Zählvariable auf Null fest. Bisher hat der Roboter auch noch keine Runde gedreht.

Der Wert dieses Wiederholungszählers muss bei jedem Schleifendurchgang um 1 erhöht werden. Daher findest du am Ende der zu wiederholenden Anweisungen den Erhöhe-Befehl für die Zählvariable: anzGemacht++;.

Die Ausführungsbedingung (anzGemacht<4) sorgt dafür, dass die Schleife genau viermal durchlaufen wird: Das erste Mal mit dem Wert 0 für anzGemacht, das zweite Mal mit dem Wert 1, das dritte Mal mit dem Wert 2 sowie das vierte und letzte Mal mit dem Wert 3.

Eine **13**-fache Wiederholung kannst du daher so schreiben:

```
int i=0;
while (i < 13) {
  // Anweisungen, die wiederholt werden
  i++; // Mitzählen!!
}
```

Statt dem Namen i könntest du auch einen Namen wie wdhZaehler benutzen oder sonst einen Namen. Informatiker verwenden gern kurz und knapp i als Zählvariable. Längere Namen blähen die Ausdrücke auf. Daher werden wir meist auch i oder j als Namen für Zählvariablen verwenden.

Wenn von Anfang an feststeht, wie oft etwas wiederholt werden muss, dann kann man Zählschleifen benutzen. Es gibt eine andere Formulierung für Zählschleifen, die wir aber nicht benutzen müssen, denn jede Wiederholung lässt sich wie gesehen mit **while** formulieren. Diese spezielle Formulierung mit dem Schlüsselwort **for** lautet:

```
for (int i=0; i<13; i++) {
  // Anweisungen, die wiederholt werden
}
```

Das Schlüsselwort for erinnert an: **für** alle Wiederholungsdurchgänge ist Folgendes zu machen.

Diese drei Schreibweisen sind gleichwertig. Nur die letzten beiden sind Java-Standard:

```
public void vierVor() {
   wdh (4 mal) {
      einsVor();
   }
}
```

```
public void vierVor() {
   for (int i=0; i<4; i++){
      einsVor();
   }
}
```

```
public void vierVor() {
   int i=0;
   while (i < 4) {
      einsVor();
      i++;
   }
}
```

Aufgaben:

1. ***Patrouille**: Teste mit dem Roboter unten links die Methode rundeDrehen() und betrachte anschließend den Quelltext. Wie oft läuft er die Strecke ab? Ändere die Methode so, dass er 10x hin und herläuft, d.h. 10x nach rechts und 10x wieder nach links.*
   Welche Zahl darf anzGemacht nicht überschreiten, damit der Roboter genau 800x hin und herläuft?
2. *Warum wird die Variable anzGemacht aus der Methode rundeDrehen() nicht im Objektinspektor angezeigt?*
3. *Verändere in der Methode rundeDrehen() die Zählschleife so, dass du anstatt einer while-Schleife eine for-Schleife verwendest. Oben stehende Tabelle kann dir als Hilfe dienen. Der Roboter soll nach Methodenaufruf genau 5x hin und herlaufen.*
4. ***Zu Befehl:** Vervollständige die Methode dreheAnzahlRunden(int anz) so, dass der Roboter anz-Runden dreht, je nach übergebenem Parameter für anz. So lässt der Aufruf dreheAnzahlRunden(7); den AB10 sieben Mal hin und herlaufen.*
   Blöderweise kann ihm der Strom ausgehen… zum Glück hat er jedoch drei Akkus dabei. Ergänze die Methode dreheAnzahlRunden(int anz) so, dass er seine Akkus sinnvoll einsetzt.
   (Tipp: mit if(getEnergie()<60) kann er überprüfen, ob sein Ladezustand ausreicht.)
5. ***Laufe x Schritte:** Implementiere eine Methode, die den Roboter genau x Schritte nach vorne laufen lässt.*
6. ***Was tut es?:** Analysiere das Verhalten der nebenstehenden Methode (Welche Aufgabe erledigt der Roboter?).*
   Entscheide, welche der Schleifen sinnvoll durch eine For-Schleife ersetzt werden kann. Implementiere die Methode mit einer for-Schleife und benenne die Methode geeignet.
   Kann man durch einen geeigneten Aufruf der Methode alle Schrauben links unten auf einmal einsammeln?

```
public void wastutes(int anz) {
 int i=0;
 while(i<anz) {
  while(!this.istWandVorne()) {
   aufnehmen();
   einsVor();
  }
  dreheUm();
  i++;
 }
}
```

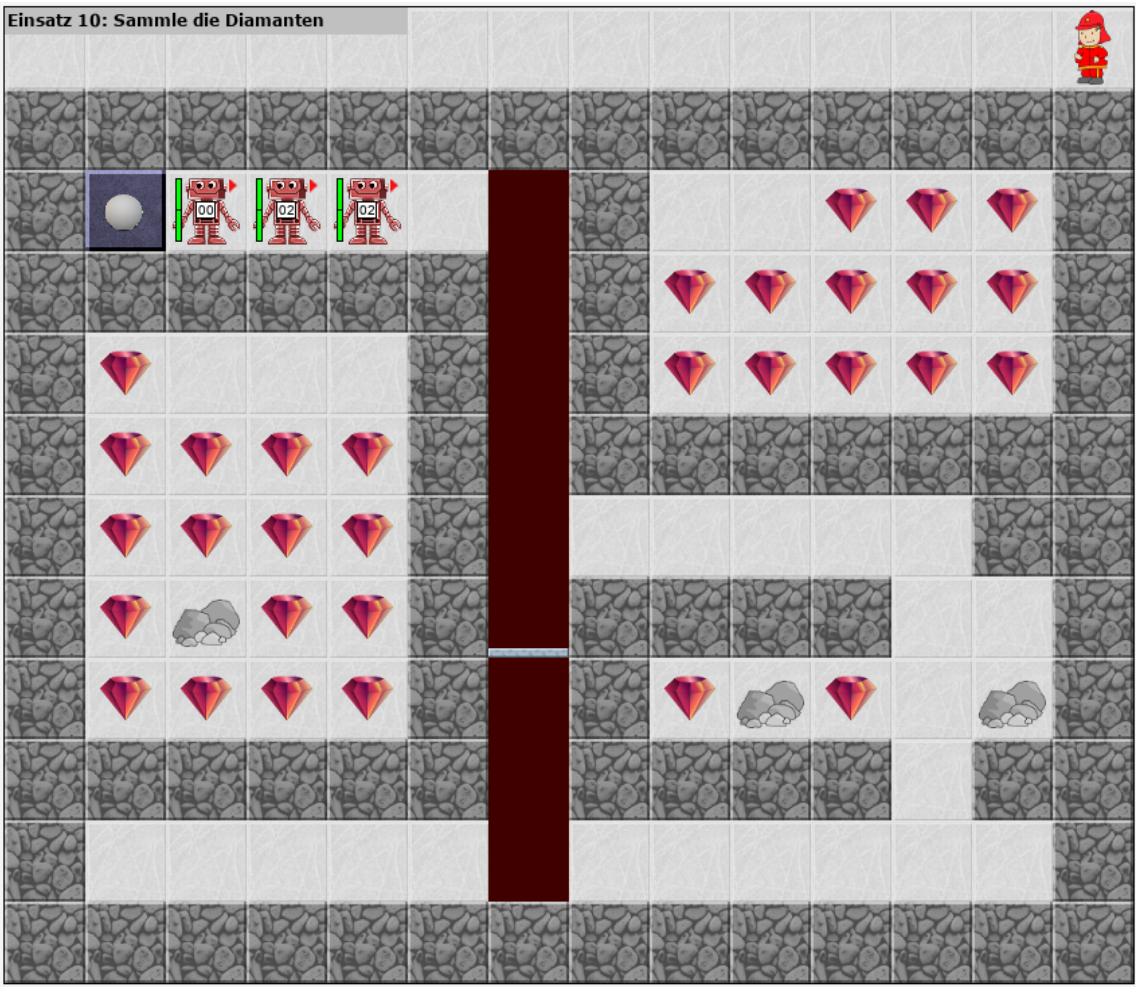
7. ***Aufräumen:** Implementiere die Methode aufraeumen(), die den Roboter rechts oben alle von oben herunterrutschende Fässer in die untere Kammer schieben lässt.*
8. ***Aufzug:** Implementiere die Methoden fahreAufzug und fahreInsStockwerk von AB9 unter Verwendung einer for-Schleife. Entscheide in beiden Fällen, ob die while-Schleife oder die for-Schleife geschickter war.*

Einsatz 10:

Die Diamantminenfirma möchte nun Ergebnisse sehen. Der ReaktorRobot soll im Bergwerk Diamanten einsammeln. Der ReaktorRobot soll dabei zunächst nur seine grundsätzliche Eignung für diese Aufgabe unter Beweis stellen:

Die Unwägbarkeiten in der Minenarbeit sind riesig. Daher ist es in diesem Szenario nicht notwendig, dass die Roboter die Aufgabe jedes Mal schaffen, sondern es reicht aus, wenn er es einmal bewältigt. Sammeln alle Roboter zusammen mind. 10 Diamanten, überlegt sich die Firma den Einsatz der Roboter, bei mind. 15, werden die Roboter (vielleicht im Fortsetzungsszenario) speziell für diese Aufgabe trainiert, bei mind. 20 Diamanten ist der Roboter schon so als Bergbauarbeiter geeignet und wird sofort gekauft.

Es stehen dir 3 Roboter (roboter1, roboter2 und roboter3) zur Verfügung. Roboter 2 und 3 führen jeweils 2 Bomben mit sich. Der Einsatzleiter muss den Einsatz koordinieren. Wie sie zu ihrem Ziel kommen, ist dabei egal. Aber Achtung: Die Steine und Diamanten sind nicht immer an der gleichen Stelle. Sprengt man Stollenwände weg, fallen die Steine und Diamanten herunter. Stürzt der Roboter runter oder wird von einem fallenden Stein getroffen, verliert er Energie.



Aufgabe:

Implementiere beim Einsatzleiter die Methode public void einsatz10(). Implementiere bei den AB10-Robotern notwendige Hilfsmethoden. Verwende dabei – wenn möglich – for-Schleifen.

Bildquellen: Die verwendeten Bilder des Roboterszenarios sind alle ohne Bildnachweis verwendbar (selbst gezeichnet, Pixabay Lizenz oder Public Domain). Genaue Nachweise: siehe *bildquellen.html*.