AB 8 – WERTZUWEISUNG + VARIABLEN TEIL 2

Immer mehr Atommüll. Wie viel passt eigentlich noch in unser Endlager? Einige Räume im Endlager sind noch frei. Dort sollen Fässer eingelagert werden. Vorher muss aber bestimmt werden, wie viel Platz dort noch ist.

Wertzuweisungen

ZIEL: *Lokale Variable als Zwischenspeicher für einen Wert kennen und einsetzen können. Lokale Variable von Objektvariablen unterscheiden können. Wertzuweisungen nutzen können.*

Aufgaben:

1. **Finde den Unterschied:** Rufe bei den beiden ***AB8***-Robotern links oben die Methoden ***zaehleSchritteBisWand1()*** bzw. ***zaehleSchritteBisWand2()*** auf. Sie sollten das gleiche Ergebnis liefern.
   Rufe danach die Methoden erneut auf. Erkläre den Unterschied.

2. **Debugging:** Über Inspect (Objektfenster) kann man sich die Werte der Attribute/Objektvariablen anzeigen lassen. Die lokalen Variablen dagegen sieht man nur im **Debugger**. Setze dazu einen Breakpoint (Stoppschild) an der das Programm unterbrochen werden soll, indem du links auf den Rand bei der Zeilennummer klickst, (hier: Zeile **19** und Zeile **30**).

```
public int zaehleSchritteBisWand1() {
    while(!istWandVorne()) {
        zaehler1++;
        einsVor();
    }
    dreheUm();
    return zaehler1;
}

public int zaehleSchritteBisWand2() {
    int zaehler2;
    zaehler2 = 0;
    while(!istWandVorne()){
        zaehler2++;
        einsVor();
    }
    dreheUm();
    return zaehler2;
}
```

   Wenn du dann die Methode aufrufst, hält das Programm am Stoppschild an und kann mit Continue aufgrund der Wiederholung bis zum nächsten Stoppschild laufen. Dabei werden sowohl Attribute (= Instance variables) als auch lokale Variablen angezeigt.

3. Entscheide, ob die folgenden Werte besser als lokale Variable oder als Attribut gespeichert werden sollten:
   a) die Farbe des Roboters,
   b) die Anzahl der insgesamt gelöschten Feuer,
   c) die Anzahl der mit dem aktuell benutzten Feuerlöscher gelöschten Feuer.

4. Entscheide, ob man bei dem Pledge-Algorithmus die Anzahl der Drehungen auch als lokale Variable hätte speichern können

5. **Summe der x- und y-Koordinate:** Rufe die Methode ***gibPositionssumme()*** an einem Roboter auf. Bewege ihn (per einsVor()- und drehen()-Befehle) an eine andere Stelle. Was meldet er jetzt als Antwort auf diesen Fragebefehl? Wie musst du ihn bewegen, damit das Ergebnis gleich bleibt? Begründe dies anhand des Quelltexts.

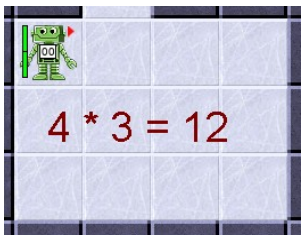
6. **Abstände:**
   a) Implementiere die Methode ***gibLaenge()*** mit einem ganzzahligen Rückgabewert: Deklariere drei lokale Variablen ***x1***, ***x2***, ***x_diff***. Weise ***x1*** den Wert des Aufrufs von ***getX()*** zu. Lasse den Roboter bis zur nächsten Wand laufen. Weise nun ***x2*** den Wert des Aufrufs von ***getX()*** zu. Berechne die Differenz von ***x2*** und ***x1*** und speichere sie in ***x_diff***. Gib diese Differenz als Ergebnis deiner Methode zurück.Teste die Methode. Stimmt das Ergebnis? **Tipp**: auch das Startfeld muss mitgezählt sein!

   b) Drehe den Roboter händisch um und rufe erneut ***gibLaenge()*** auf, sodass er anschließend wieder von rechts nach links zurückläuft. Was ist das Ergebnis? Erkläre das Ergebnis! Erweitere die Methode, sodass das Ergebnis immer positiv ist.

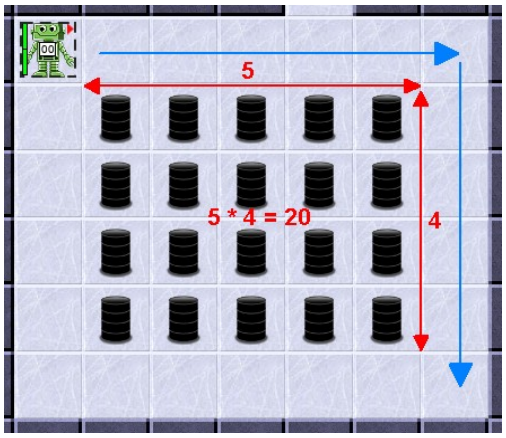
   c) Drehe den Roboter, sodass er nach unten (bzw. oben schaut). Ruf die Methode ***gibLaenge()*** auf. Was ist das Ergebnis? Erkläre das Ergebnis.
   Erweitere die Methode, sodass auch die y-Koordinaten-Abstände gezählt und ebenfalls keine negativen Ergebnisse zurückgegeben werden.

   d) Eine Methode kann immer nur einen Wert zurückgeben. Überprüfe anhand von ***x_diff*** und ***y_diff*** in welche Richtung der Roboter eigentlich gelaufen ist und gib diesen Wert zurück.

7. **Fläche:** Implementiere eine Methode, die die Fläche eines Raumes (Breite * Höhe) bestimmt und zurückgibt. Du kannst davon ausgehen, dass der Roboter in einer Ecke des Raumes steht.



8. **Fässer zählen:** Implementiere eine Methode, die zurückgibt, aus wie vielen Fässern ein rechteckiges Areal voll von Fässern besteht. Hierbei ist es sinnvoll sich zwei Hilfsmethoden zu schreiben:
   - die Methode ***istFassRechts()*** überprüft, ob rechts vom Roboter ein Fass ist. (siehe AB5)
   - Damit kann man eine Methode ***gibLaengeFassreihe()*** implementieren, die vorwärts läuft solange rechts ein Fass ist und dabei die Anzahl der Fässer mitzählt.
   - Die Methode ***gibAnzahlFaesser()*** bestimmt dann die Anzahl der Fässer im Rechteck.



9. Gib an, welchen Wert die Variable summe hat, wenn folgende fünf Zeilen ausgeführt wurden:

```
int summe;
summe = 10;
summe = summe - 2;
summe = summe * 4;
summe = summe + 6;
```

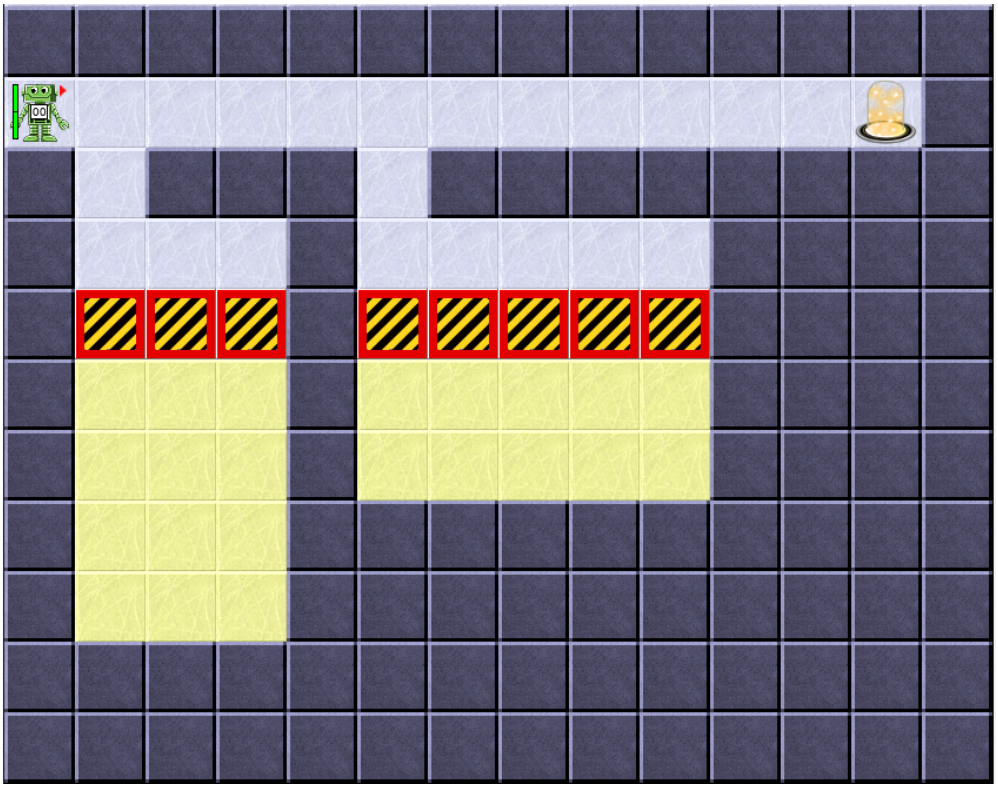
10. **Sammler:** Implementiere eine Methode, die den Roboter bis zur nächsten Wand laufen lässt und dabei alles aufhebt, was auf dem Boden liegt. Dabei soll der Wert der gesammelten Gegenstände ermittelt werden. Jede Schraube ist dabei 20 Cent Wert, jeder Akku 2,40 Euro, jeder Schlüssel 5 Euro, jeder Feuerlöscher 40 Euro. Der Wert soll dann mittels einer get-Methode jederzeit abgefragt werden können.
Tipp: Überlege dir, ob du eine lokale Variable oder ein Attribut für den Wert verwenden möchtest.

11. **Finanzamt:** Der Wert aller gesammelter Gegenstände (aus vorheriger Aufgabe) muss besteuert werden, d. h. für jeden gefundenen Gegenstand, der verkauft wird, muss die Mehrwertsteuer hinzugerechnet und ans Finanzamt abgeführt werden. Daher soll der Roboter mit ***getMehrwertsteuer()*** den Steuerbetrag der gefundenen Gegenstände berechnen können. Implementiere eine Methode, die die Steuer berechnet und zurückgibt.

Einsatz 8: Bestimme die Lagerkapazität des Endlagers

Einige Räume im Endlager sind noch frei. Dort sollen Fässer eingelagert werden. Vorher muss aber bestimmt werden, wie viel Platz dort noch ist. Die Räume gehen alle nach rechts von dem oberen Gang ab. Man kommt immer am linken oberen Eck in den Raum. Die ersten zwei Zeilen müssen frei bleiben, damit man mit dem Gabelstapler rangieren kann. Daher können die Fässer nur auf den gelben Feldern platziert werden.

Auftrag: Bestimme die Anzahl der gelben Felder und gehe zum Portal. Der Leveltest (also die Methode einsatz8()) muss als Ergebnis die Anzahl der gelben Felder zurückgeben.



Tipps:
- starte den Einsatz 8 mehrmals, um die einen Überblick über den Aufbau der Räume zu verschaffen. Die Anzahl und Größe der Räume ist zufällig.
- benutze die im AB8 erstellten Methoden. Überlege dir hierfür an welche Position der Roboter in die Räume gebracht werden muss, sodass man die zuvor erstellten Methoden sinnvoll nutzen kann.

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