

Aufgabe:

Version 1: KARA steht vor einer Reihe von Kleeblättern, an deren Ende ein Baum steht. Die Kleeblattreihe kann Lücken haben. KARA soll alle Kleeblätter aufsammeln.

Version 2: KARA soll die Kleeblattreihe invertieren, d.h. wenn er auf einem Kleeblatt steht, soll er es aufnehmen; wenn er keines vorfindet, soll er eines ablegen.

Das Neue: KARA soll in Abhängigkeit von einer Bedingung etwas wiederholt ausführen. Hier: *Solange kein Baum vor ihm steht*, soll er etwas tun, nämlich Kleeblätter sammeln.

```
import JavaKaraProgram;
public class KleeblattSammeln extends JavaKaraProgram {
// Anfang von KleeblattSammeln
    public void myProgram()
    { // Anfang von myProgram
        if (kara.onLeaf()) { kara.removeLeaf(); }
        while (!kara.treeFront())
        {
            kara.move();
            if (kara.onLeaf()) { kara.removeLeaf(); }
        }
    } // Ende von myProgram
} // Ende von KleeblattSammeln
```

Erläuterungen:

1. Die Syntax der while-Schleife:

```
while ( Bedingung )
{ Auszufuehrende Methoden, wenn ja }
```

Der Ablauf: Zuerst wird die Bedingung ausgewertet. Wenn das Ergebnis „ja“ lautet, werden die Methoden, die in { } eingeschlossen sind, ausgeführt. Nach der Ausführung wird die Bedingung wieder ausgewertet, usw. Wird die Bedingung nicht erfüllt, macht das Programm mit der nächsten Anweisung nach dem in { } eingeschlossenen Block weiter.

Wird die Bedingung schon beim ersten Mal nicht erfüllt, wird der Anweisungsblock kein einziges Mal ausgeführt.

2. Von der while-Schleife gibt es eine seltener genutzte Form, die do-Schleife. Die Syntax der do-Schleife:

```
do {
    Auszufuehrende Methoden
} while ( Bedingung );
```

Der Unterschied zwischen den beiden Schleifenformen besteht im Zeitpunkt der Prüfung der Bedingung. Bei der `do`-Schleife findet die Prüfung am Ende der Schleife statt. Sie wird also mindestens einmal durchlaufen. Das kann gefährlich sein. Beachte das Semikolon am Ende der Zeile mit `while`!

Fragen:

1. Wozu dient die folgende Zeile, die vor der `while`-Schleife steht? Für welche Situation ist sie notwendig?

```
if (kara.onLeaf()) { kara.removeLeaf(); }
```

2. Die folgende Lösung ist für eine bestimmte Situation nicht korrekt. Welche?

```
import JavaKaraProgram;
public class KleeblattSammeln extends JavaKaraProgram {
// Anfang von KleeblattSammeln
    public void myProgram()
    { // Anfang von myProgram
        while (!kara.treeFront())
        {
            if (kara.onLeaf()) { kara.removeLeaf(); }
            kara.move();
        }
    } // Ende von myProgram
} // Ende von KleeblattSammeln
```

Verbessere das Programm durch Einfügen einer zusätzlichen `if`-Anweisung.

3. Was ist hier falsch?

```
import JavaKaraProgram;
public class KleeblattSammeln extends JavaKaraProgram {
// Anfang von KleeblattSammeln
    public void myProgram()
    { // Anfang von myProgram
        while (!kara.treeFront() && kara.onLeaf())
        {
            kara.removeLeaf();
            kara.move();
        }
    } // Ende von myProgram
} // Ende von KleeblattSammeln
```

4. Schreibe das Programm mit einer `do`-Schleife.
5. `while (true) { kara.turnLeft(); }` Was passiert?
6. `while (!kara.treeFront()); { kara.move(); }` Was passiert und warum?