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.

```
1  import javakara.JavaKaraProgram;
2  public class KleeblattSammeln extends JavaKaraProgram
3  {  // Anfang von KleeblattSammeln
4    public void myProgram()
5    {  // Anfang von myProgram
6    if (kara.onLeaf()) { kara.removeLeaf(); }
7    else { kara.putLeaf(); }
8    while (!kara.treeFront())
9      {
10       kara.move();
11       if (kara.onLeaf()) { kara.removeLeaf(); }
12       else { kara.putLeaf(); }
13     }
14   }  // Ende von myProgram
15 } // Ende von KleeblattSammeln
```

Fragen:

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

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

   Antwort: In der `while`-Schleife geht KARA zuerst einen Schritt vor. Also vergisst er eventuell ein Blatt, auf dem er startet.

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

```
1  import javakara.JavaKaraProgram;
2  public class KleeblattSammeln extends JavaKaraProgram
3  {  // Anfang von KleeblattSammeln
4    public void myProgram()
5    {  // Anfang von myProgram
6    while (!kara.treeFront())
7      {
8        if (kara.onLeaf()) { kara.removeLeaf(); }
9        kara.move();
10     }
11   }  // Ende von myProgram
12 } // Ende von KleeblattSammeln
```

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

Antwort: Wenn KARA auf dem Feld vor dem Baum angekommen ist, wird die Schleife nicht mehr ausgeführt. Dann bleibt das letzte Blatt liegen. Also muss man nach der Schleife noch die folgende Zeile einfügen:

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

3. Was ist hier falsch?

```
1  import javakara.JavaKaraProgram;
2  public class KleeblattSammeln extends JavaKaraProgram
3  {  // Anfang von KleeblattSammeln
4    public void myProgram()
5    {  // Anfang von myProgram
6    while (!kara.treeFront() && kara.onLeaf())
7      {
8        kara.removeLeaf();
9        kara.move();
10     }
11   }  // Ende von myProgram
12 } // Ende von KleeblattSammeln
```

Antwort: Die Schleife wird nur ausgeführt, wenn vorne kein Baum ist *und* ein Blatt vorhanden ist. Steht KARA auf einem leeren Feld, wird nichts mehr getan und KARA bleibt dort stehen.

4. Schreibe das Programm mit einer `do`-Schleife.

```
1  import javakara.JavaKaraProgram;
2  public class KleeblattSammeln extends JavaKaraProgram
3  {  // Anfang von KleeblattSammeln
4    public void myProgram()
5    {  // Anfang von myProgram
6    do
7      {
8        if (kara.onLeaf()) { kara.removeLeaf(); }
9        if (!kara.treeFront()) { kara.move(); }
10     } while (!kara.treeFront());
11     if (kara.onLeaf()) { kara.removeLeaf(); }
12   }  // Ende von myProgram
13 } // Ende von KleeblattSammeln
```

Die `do`-Schleife ist gefährlich! Für den ersten Durchlauf muss also ein

```
if (!kara.treeFront()) { kara.move(); }
```

eingefügt werden. KARA könnte ja beim Start schon direkt vor einem Baum stehen.

5. `while (true) { kara.turnLeft(); }` Was passiert?

Antwort: Die Bedingung ist immer *wahr*, also wird die Schleife immer wieder ausgeführt. Es entsteht eine Endlosschleife.

6. `while (!kara.treeFront()); { kara.move(); }` Was passiert und warum?

   Antwort: Ein kleiner Fehler mit großer Wirkung! Die geschweiften Klammern für den Anweisungsblock der Schleife darf man weglassen, wenn der Block aus einer einzigen Anweisung besteht. Richtig wäre demnach:

   ```
   while (!kara.treeFront()) kara.move();
   ```

   Da hier aber ein Semikolon hinter der Bedingungsklammer steht, sieht der Compiler die Schleife, die keine Anweisung enthält, als beendet. Man erhält eine Endlosschleife und `kara.move();` wird niemals ausgeführt. Man hat versehentlich Folgendes programmiert:

   ```
   while (!kara.tree.Front()) /* Mache gar nichts! */ ;
   { kara.move();}            // Gehoert nicht mehr zur Schleife
   ```