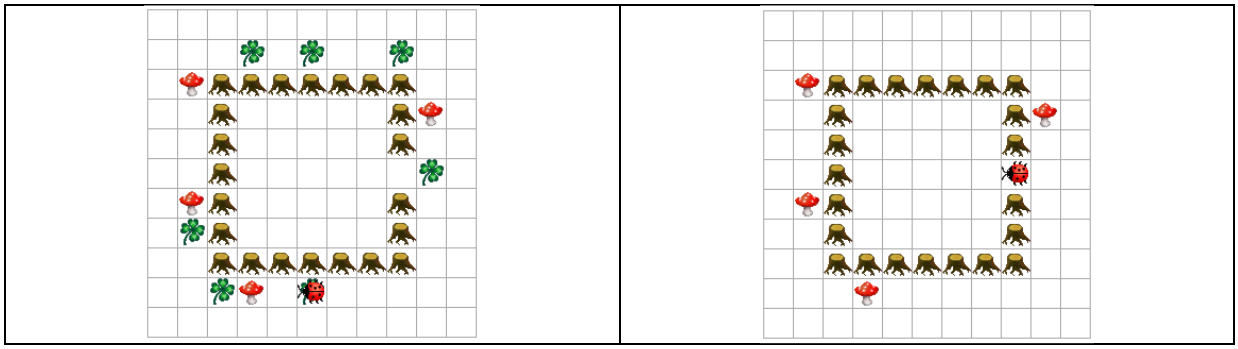
AUFGABE 1: PROGRAMM SCHREIBEN (6 PUNKTE)

Kara soll den Eingang in das Baum-Rechteck suchen, indem er im Uhrzeigersinn das Rechteck abläuft:



Unterwegs soll er alle Kleeblätter fressen. Um die Pilze kann er einfach herum laufen; sie stehen so, dass er problemlos herum laufen kann.

Schreiben Sie ein Programm, das Kara zum Eingang laufen lässt!

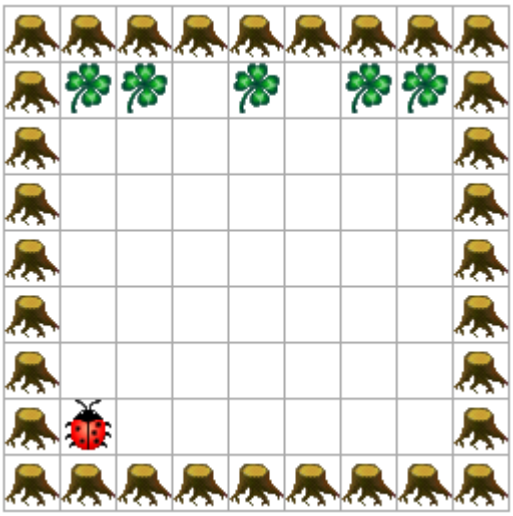
```
public void myMainProgram() {




}
```

AUFGABE 2: PROGRAMME LESEN (6 PUNKTE)

AUFGABE 2.1 (3 PUNKTE)

Sie erhalten ein Programm zur Analyse, in dem dummerweise ausgerechnet die zentrale Methode keinen sprechenden Namen erhalten hat… da hilft nur, die Programmausführung zu simulieren!

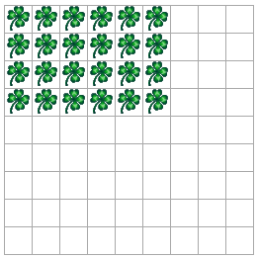
Zeichnen Sie in die Welt links ein, wie die Welt nach der Ausführung des Programms rechts aussieht. Markieren Sie insbesondere auch deutlich, wo Kara steht und in welche Richtung er schaut:



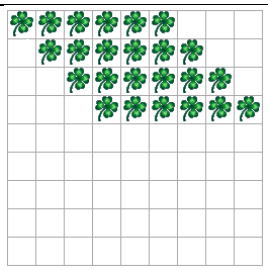
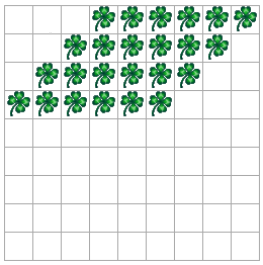
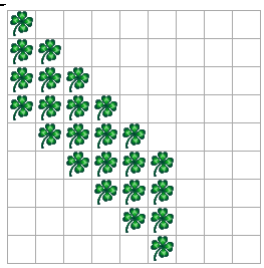
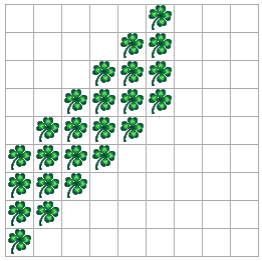
```
public void myMainProgram() {
	while (!kara.treeRight()) {
		irgendsoeinemethode();
		zurNaechstenSpalte();
	}
	irgendsoeinemethode();
}

void irgendsoeinemethode() {
	laufeZuBaum();
	umdrehen();

	if (kara.onLeaf()) {
		laufeZuBaum();
		kara.putLeaf();
	} else {
		laufeZuBaum();
	}
}

void umdrehen() {
	kara.turnRight();
	kara.turnRight();
}

void laufeZuBaum() {
	while (!kara.treeFront()) {
		kara.move();
	}
}

void zurNaechstenSpalte() {
	kara.turnLeft();
	kara.move();
	kara.turnLeft();
}
```

AUFGABE 2.2 (3 PUNKTE)

Es war einmal ein ganz gewöhnliches Rechteck aus Kleeblättern…



… das traf auf vier verschiedene Programm, die es in allen Himmelsrichtungen arg verzerrten:

1 	2 
3 	4 

Hier sind die vier Programme:

A

```
for (int y = 0; y < 4; y++) {
  for (int x = 3-y; x < 9-y; x++) {
    world.setLeaf(x, y, true);
  }
}
```

B

```
for (int x = 0; x < 6; x++) {
  for (int y = 5-x; y < 9-x; y++) {
    world.setLeaf(x, y, true);
  }
}
```

C

```
for (int x = 0; x < 6; x++) {
  for (int y = x; y < x + 4; y++) {
    world.setLeaf(x, y, true);
  }
}
```

D

```
for (int y = 0; y < 4; y++) {
  for (int x = y; x < y + 6; x++) {
    world.setLeaf(x, y, true);
  }
}
```

Ordnen Sie jedes der vier verzerrten Rechtecke demjenigen Programm, welches das verzerrte Rechteck (in einer zu Beginn leeren Welt) erzeugt:

1 ⇔	2 ⇔	3 ⇔	4 ⇔

AUFGABE 3: JAVAKARA-PROGRAMM LESEN (6 PUNKTE)

Sie erhalten ein JavaKara-Programm, das ein anderer Programmierer geschrieben hat:

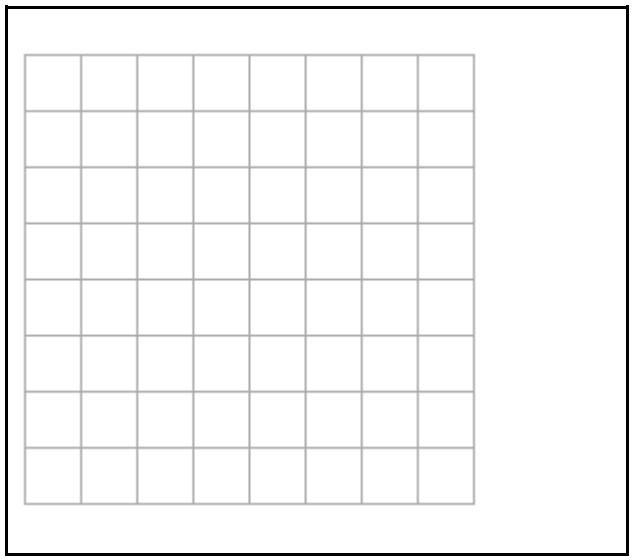
```
public void myMainProgram() {
      int halbeHoehe = world.getSizeY() / 2;
      for (int x = 0; x < world.getSizeX() / 2; x++) {
            int anzahl = tools.intInput(
                  "Geben Sie eine Zahl zwischen 0 und " +
                  halbeHoehe + " ein.");
            legeBlaetter(x, anzahl);
            legeBlaetter(world.getSizeX() - 1 - x, anzahl);
      }
}

void legeBlaetter(int x, int anzahl) {
      for (int y = 0; y < anzahl; y++) {
            world.setLeaf(x, y, true);
            world.setLeaf(x, world.getSizeX() - 1 - y, true);
      }
}
```

Zur Erinnerung: world.getSizeY() liefert die Anzahl Zeilen der Welt; world.getSizeX() liefert die Anzahl Spalten der Welt; tools.intInput() zeigt den Text an und erfragt eine Ganzzahl vom Benutzer.

Immerhin hat er Ihnen auch den Hinweis hinterlassen, dass das Programm gedacht ist für anfänglich leere Welten, deren Breite eine gerade Zahl (2, 4, 6, ...) sein sollte.

AUFGABE 3.1 (3 PUNKTE)

Zeichnen Sie in die leere Welt unten, wie die Welt **nach** Ausführung des Programmes aussieht. Beachten Sie, dass die Welt 8x8 Felder gross ist. **Der Benutzer gibt nacheinander die Zahlen 3,2,1,2 ein:**



AUFGABE 3.2 (2 PUNKTE)

Beschreiben Sie in je zwei, drei Sätzen präzise, was welche Methode macht:

legeBlaetter(...)	
myMainProgram()	

AUFGABE 3.3 (1 PUNKT)

Beschreiben Sie in ein, zwei Sätzen, was im Hauptprogramm die zweifachen Aufrufe von legeBlaetter(...) bewirken:

AUFGABE 4: PROGRAMMIEREN ERKLÄREN (6 PUNKTE)

Stellen Sie sich vor, Sie möchten einem Kollegen erzählen, was Sie über das Programmieren gelernt haben. Ihr Kollege hat noch keine Erfahrung im Programmieren. Erklären Sie ihm daher anhand von einfachen JavaKara-Beispielen, wie der Computer ein JavaKara-Programm ausführt. Verwenden Sie dazu das folgende Beispiel, das wir auch im Unterricht betrachtet haben:

```
5   public void myProgram() {
6     ganzeDrehung();
7   }
8
9   void ganzeDrehung() {
10    dreiViertelDrehung();
11    viertelDrehung();
12  }
13
14  void dreiViertelDrehung() {
15    halbeDrehung();
16    viertelDrehung();
17  }
18
19  void viertelDrehung() {
20    kara.turnLeft();
21    kara.move();
22    kara.turnRight();
23    kara.move();
24    kara.turnRight();
25  }
26
27  void halbeDrehung() {
28    viertelDrehung();
29    viertelDrehung();
30  }
```

Erklären Sie ihrem Kollegen anschaulich, wie der Computer dieses Programm liest und ausführt:

- Was bedeutet zum Beispiel void ganzeDrehung() in Zeile 9?
- Was bedeutet zum Beispiel viertelDrehung() in Zeile 11?
- Woher weiss der Computer, welcher Befehl als nächstes ausgeführt werden muss?
- Wo startet und endet die Programmausführung?