

## AUFGABE 1: REGULÄRE AUSDRÜCKE (5 PUNKTE)

a	Das Zeichen a
.	Ein beliebiges Zeichen
[abc]	Ein beliebiges Zeichen aus der Menge {a, b, c}
[a-f]	Ein beliebiges Zeichen aus der Menge {a, b, c, d, e, f}
\d	eine Ziffer [0-9]
X Y	X oder Y
X*	Eine beliebige Wiederholung von X
X+	Mindestens einmal X
X?	Höchstens einmal X
{n}	der vorangehende Ausdruck muss exakt n mal vorkommen
{m,n}	der vorangehende Ausdruck muss mindestens m mal und darf höchstens n mal vorkommen
(xyz)	Gruppierung: xyz müssen miteinander vorkommen
[^xyz]	Nicht x, nicht y, nicht z
\X	X ein Sonderzeichen \()\[\]*+?{}\.^\$-

### AUFGABE 1.1 (3 PUNKTE)

Sie möchten Uhrzeiten mit regulären Ausdrücken beschreiben. Sie haben bereits mit Google gesucht und einige mögliche Kandidaten ermittelt. Prüfen Sie, ob die folgenden regulären Ausdrücke korrekt Uhrzeiten im Vierundzwanzigstunden-Format beschreiben oder nicht. Die Uhrzeiten in diesem Format gehen von 00:00 bis zu 23:59.

Regulärer Ausdruck	Kurze Begründung, warum der Ausdruck korrekt ist <b>ODER</b> jeweils zwei Gegenbeispiele, die zeigen, warum der Ausdruck nicht korrekt ist
[012][0-9]:[0-5][0-9]	
([01]\d 2[0-3]):([0-5]\d)	

(Aufgabe 1.1, Fortsetzung)

Regulärer Ausdruck	Begründung, warum der Ausdruck korrekt ist <b>ODER</b> jeweils zwei Gegenbeispiele, die zeigen, warum der Ausdruck nicht korrekt ist
<code>\d\d:\d\d</code>	
<code>([01][0-4]   [01][5-9]   2[0-3]):([0-5][0-9])</code>	

#### AUFGABE 1.2 (2 PUNKTE)

Postleitzahlen in England haben folgendes Format (etwas vereinfacht):

- 1-2 Grossbuchstaben für das Sortieramt, z.B. G für Glasgow
- 2 Ziffern für den Postbezirk
- 1 Leerzeichen
- 1 Ziffer für das Zustellamt
- 2 Buchstaben für den Zustellbereich eines Briefträgers

Ein Beispiel einer gültigen Postleitzahl in diesem Format ist RH10 2RY.

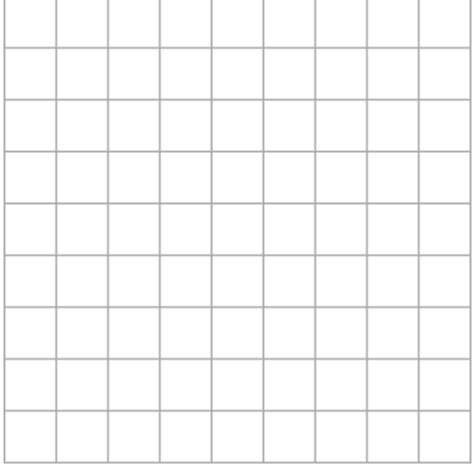
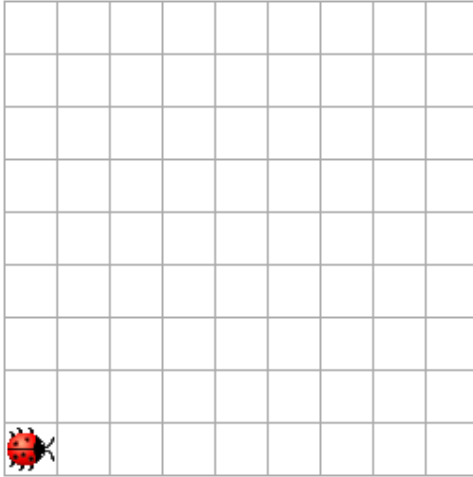
**Geben Sie einen regulären Ausdruck an, die die beschriebenen Postleitzahlen erkennt:**

--

## AUFGABE 2: PROGRAMM LESEN – PROGRAMM ERGÄNZEN UND LESEN (5 PUNKTE)

### AUFGABE 2.1 (3 PUNKTE)

Zeichnen Sie in die Welten links ein, wie die Welten nach der Ausführung des Programms rechts aussehen.

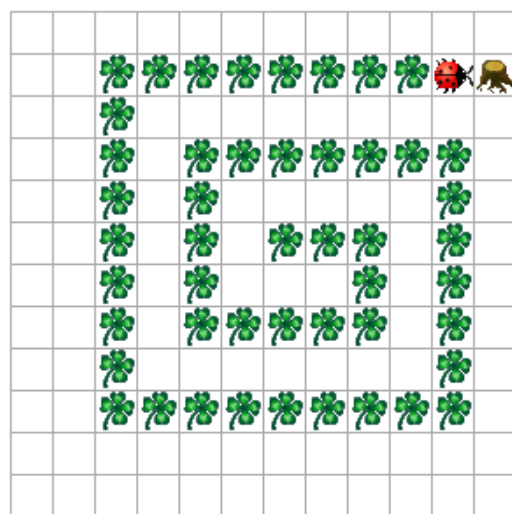
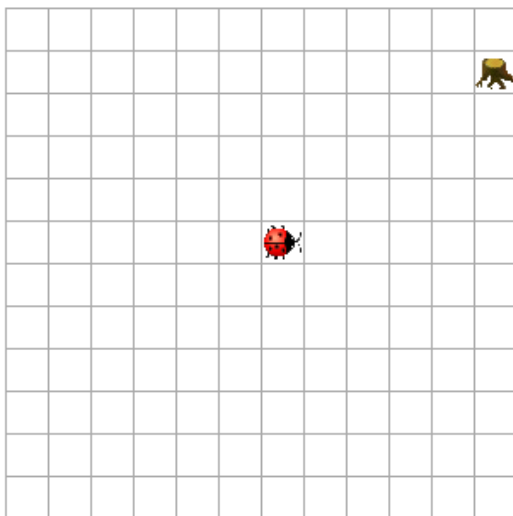
	<pre> for (int i = 1; i &lt; world.getSizeX() - 1; i++) {     world.setLeaf(i, i, true); }  for (int i = 1; i &lt; world.getSizeX() - 1; i++) {     world.setLeaf(i, world.getSizeY() - 1 - i, true); }  /* Zur Erinnerung: world.setLeaf(x, y, true) legt an der Koordinate (x, y) ein Kleeblatt, wenn dort noch keines liegt. world.getSizeX() gibt die Breite der Welt zurück, world.getSizeY() die Höhe. */ </pre>
	<pre> public void myProgram() {     treppe(4); }  void treppe(int anzahlStufen) {     for (int i = 0; i &lt; anzahlStufen; i++) {         stufe(2);     } }  void stufe(int laenge) {     legeX(laenge);     kara.turnLeft();     legeX(laenge);     kara.turnRight(); }  void legeX(int laenge) {     for (int i = 0; i &lt; laenge; i++) {         kara.putLeaf();         kara.move();     } } </pre>

## AUFGABE 2.2 (2 PUNKTE)

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

```
while (!kara.treeFront()) {  
    if (kara.mushroomFront()) {  
        kara.turnRight();  
    }  
    kara.putLeaf();  
    kara.move();  
}
```

Markieren Sie in der Welt unten links diejenigen Felder, auf denen Sie Pilze setzen müssen, damit das obige Programm Kara die Spirale in der Welt unten rechts legen lässt.



### AUFGABE 3: JAVA-KARA-PROGRAMM LESEN (5 PUNKTE)

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

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

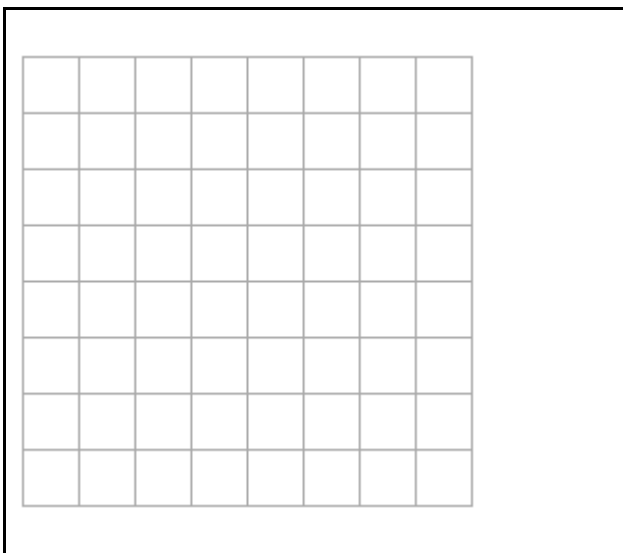
void legeBlaetter(int y, int anzahl) {
    int halbeBreite = world.getSizeX() / 2;
    for (int x = halbeBreite - anzahl; x < halbeBreite + anzahl; x++) {
        world.setLeaf(x, y, true);
    }
}
```

Zur Erinnerung: world.getSizeY() liefert die Anzahl Zeilen der Welt; world.getSizeX() liefert die Anzahl Spalten der Welt.

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 (2 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 1,2,3,2 ein:**



### AUFGABE 3.2 (2 PUNKTE)

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


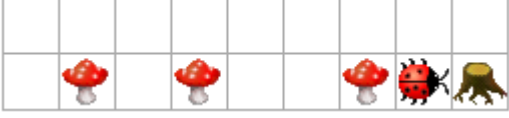


<b>legeBlaetter(...)</b>	
<b>myMainProgram()</b>	

### AUFGABE 3.3 (1 PUNKT)

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

#### AUFGABE 4: JAVAKARA-PROGRAMM SCHREIBEN (5 PUNKTE)

Schreiben Sie ein Programm, das Kara um Pilze laufen und dabei Kleeblätter aufnehmen lässt wie in der Abbildung unten. Er soll dabei alle Kleeblätter in der unteren Zeile essen. Er ist fertig, wenn er vor dem Baum steht:

 <p>Beispiel 1: Welt vorher</p>	 <p>Beispiel 1: Welt nachher</p>
 <p>Beispiel 2: Welt vorher</p>	 <p>Beispiel 2: Welt nachher</p>

**Hinweis:** Der Einfachheit halber nehmen wir an, dass nie ein Pilz direkt vor einem Baum steht. Wir nehmen ferner an, dass nie zwei Pilze direkt hintereinander stehen.

#### AUFGABE 4.1 (4 PUNKTE)

Der Anfang ist bereits gemacht, Sie können die Methoden `entferneKleeblatt()` und `laufeUmPilz()` verwenden, wenn Sie möchten, und sich auf das Hauptprogramm in `myMainProgram()` konzentrieren. Schreiben Sie ein Programm, das Kara zu dem Baum laufen lässt und das dabei alle Kleeblätter entfernt:

```
public void myMainProgram() {
```

```

}

void entferneKleeblatt() {
    if (kara.onLeaf()) {
        kara.removeLeaf();
    }
}

void laufeUmPilz() {
    kara.turnLeft();
    kara.move();
    kara.turnRight();
    kara.move();
    kara.move();
    kara.turnRight();
    kara.move();
    kara.turnLeft();
}

```

#### AUFGABE 4.2 (1 PUNKT)

Beschreiben Sie in zwei, drei Sätzen, was an dem Programm geändert werden müsste, wenn mehrere Pilze nebeneinander stehen dürften wie in der Abbildung unten?



Mehrere Pilze nebeneinander



## AUFGABE 5: PROGRAMMIEREN ERKLÄREN (5 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 Sie den Computer dazu bringen, JavaKara Dinge tun zu lassen.

Erklären Sie ihm zunächst umgangssprachlich, was Sie JavaKara tun lassen möchten. Verwenden Sie dazu ein möglichst einfaches Beispiel: „Stell' Dir vor, ich will den Marienkäfer dazu bringen, 33 Kleeblätter zu legen. Mit den Befehlen `kara.putLeaf();` `kara.move();` legt der Käfer ein einzelnes Kleeblatt und läuft weiter. Aber ich möchte diese Befehle natürlich nicht 33 mal kopieren müssen! Also verwende ich eine sog. Zählschleife, um den Computer die Befehle 33 mal ausführen zu lassen.“

Schreiben Sie anschliessend auf, wie Ihr Beispiel in Java umgesetzt werden würde: „Das würde ich in Java wie folgt schreiben: `for (int i = 0; i < 33; i++) { kara.putLeaf(); kara.move(); }`“. Die Feinheiten des Codes (hier zum Beispiel Zählbeginn bei 0) brauchen Sie dabei nicht weiter zu vertiefen.

**Erklären Sie Verzweigungen (if-else): Wie kann JavaKara Entscheidungen treffen?**

**Erklären Sie Wiederholungen (while): Wie kann JavaKara Dinge wiederholen?**