Day 6: Guard Gallivant

Tag 6 ist eine klassische Aufgabe, die man gut in einem zweidimensionalen Array lösen kann. Besonders Teil 1 ließe sich theoretisch auch ohne einem Array lösen, indem man nur die Koordinaten der Hindernisse miteinander vergleicht/verrechnet und sich so von Hindernis zu Hindernis hangelt. Mit dem Array geht es aber intuitiver.

Teil 1

Vorgehensweise:

- Übertrage die Eingabedatei in ein zweidimensionales char-Array. Merke dir dabei, an welcher Koordinate der Guard losläuft.
- Erschaffe eine Richtungs-Variable vom Typ int. Wenn der Guard anschließend beim Marschieren auf ein Hinderniss trifft und abbieben muss, dann kannst du die int-Zahl immer um eins erhöhen, was einer neuen Richtung entspricht. Da es nur 4 Richtungen gibt, muss die Zahl nach dem Erhöhen u. U. automatisch auf 0 gesetzt werden, sodass die Werte 0, 1, 2 und 3 als Richtung zulässig sind (Tipp: Modulo).
- Beginne anschließend eine Endlosschleife (`while(true){…}`), um den Guard so lange laufen zu lassen, bis er aus der Map herausläuft. Anders darf die Schleife nicht abgebrochen werden.
    - Speichere, auf welche neue Koordinate der Guard als nächstes geradeaus laufen **würde**. Hier müssen für die 4 Richtungen 4 Fälle beachtet werden.
        - Würde der Guard nun in dieser neuen Koordinate in einem Hindernis stehen, so bewegt er sich nicht auf die neue Koordinate, sondern ändert hingegen seine Richtung-Variable.
        - Ansonsten aktualisiert man die aktuelle Position vom Guard und markiert außerdem die Position auf der Map (z. B. mit char 's' für **s**tep). In diesem Moment darf man auch die Anzahl der zurückgelegten Schritte um 1 erhöhen - allerdings nur dann, wenn die Position nicht bereits zuvor mit einem 's' markiert war.
    - Am Ende der Endlosschleife muss man noch prüfen, ob sich der Guard nun direkt am Rand der Map befindet - nur genau dann kann man die Endlosschleife mit `break;` abbrechen.

Lösungsvorschlag

```
public void partOne() {
    int width = inputLines.get(0).length();
    int height = inputLines.size();

    // um später darin die Startposition vom Guard zu speichern
    int guardX = 0;
    int guardY = 0;

    // Karte
    char[][] map = new char[width][height];

    //speichere jede Koordinate im Array
```

```
    for (int y = 0; y < height; y++) {
        String line = inputLines.get(y);
        for (int x = 0; x < width; x++) {
            // nimm das aktuelle Zeichen
            char c = line.charAt(x);
            if (c == '^') {
                map[x][y] = 's';

                // merk dir, dass hier der Guard losläuft
                guardX = x;
                guardY = y;
            }
            else {
                map[x][y] = line.charAt(x);
            }
        }
    }

    int steps = 1;
    int direction = 0;
    while(true) {
        int nextX, nextY;
        if (direction == 0) { // up
            nextX = guardX;
            nextY = guardY-1;
        } else if (direction == 1) { // right
            nextX = guardX+1;
            nextY = guardY;
        } else if (direction == 2) {// down
            nextX = guardX;
            nextY = guardY+1;
        } else { //left
            nextX = guardX-1;
            nextY = guardY;
        }

        // Wenn er sich nun in einem Hindernis befinden würde, dann soll er
abbiegen.
        if (map[nextX][nextY] == '#') {
            direction = (direction + 1) % 4;
        } else { // Andernfalls läuft er nach vorne und markiert seinen Weg
            guardX = nextX;
            guardY = nextY;
            if (map[guardX][guardY] != 's') {
                steps++;
            }
            map[guardX][guardY] = 's';
        }
```

```
        // Prüfe, ob der Guard direkt am Rand der Map steht.
        if (guardX == 0 || guardX == width - 1 || guardY == 0 || guardY ==
height - 1) {
            break;
        }
    }

    System.out.println(steps);
}
```

Teil 2

Teil 2 macht manches insofern etwas schwerer, als dass man sich nun zum Einen merken muss, an welcher Stelle man bereits **in welcher Richtung** entlang gelaufen ist um Loops zu erkennen. Zum Anderen muss man in einer weiteren Schleife das Hindernis jedes Mal an eine neue Stelle setzen und gleichzeitig die vorherigen Markierungen löschen bzw. eine Startvorlage der Karte kopieren.

Lösungsvorschlag

```
public void partTwo() {
    int width = inputLines.get(0).length();
    int height = inputLines.size();

    int startX = 0;
    int startY = 0;

    // Kopiere den Input in ein zweidimensionales Array
    char[][] originalMap = new char[width][height];
    for (int y = 0; y < height; y++) {
        String line = inputLines.get(y);
        for (int x = 0; x < width; x++) {
            char c = line.charAt(x);
            if (c == '^') {
                // Speichere die Richtung, in die sich der Guard zu Beginn
bewegt.
                originalMap[x][y] = (char)(0+'0'); // Die Schreibweise sorgt
dafür, dass die Zahl 0 als lesbarer char 0 abgespeichert wird.
                startX = x;
                startY = y;
            }
            else {
                originalMap[x][y] = c;
            }
        }
    }

    int direction = 0;
    int obstructions = 0;  // zähle die möglichen neuen Hindernis-
Koordinaten
```

```
    for (int x = 0; x < width; x++) {
        for (int y = 0; y < height; y++) {
            char[][] map = new char[width][height];
            for (int mx = 0; mx < width; mx++) {
                for (int my = 0; my < height; my++) {
                    map[mx][my] = originalMap[mx][my];
                }
            }

            // Setze die Richtung des Guards zu Beginn auf 0
            direction = 0;

            // Das Hindernis darf weder auf ein anderes Hindernis noch auf die Startposition gesetzt werden.
            if (map[x][y] == '#' || (startX == x && startY == y)) {
                continue;
            }

            // Platziere das Hindernis
            map[x][y] = '#';

            // Platziere den "imaginären" Guard auf die Startposition
            int guardX = startX;
            int guardY = startY;

            while(true) {
                int nextX, nextY;
                if (direction == 0) { // up
                    nextX = guardX;
                    nextY = guardY-1;
                } else if (direction == 1) { // right
                    nextX = guardX+1;
                    nextY = guardY;
                } else if (direction == 2) {// down
                    nextX = guardX;
                    nextY = guardY+1;
                } else { //left
                    nextX = guardX-1;
                    nextY = guardY;
                }

                // wenn ich in ein Hindernis reinlaufe, dann abbiegen
                if (map[nextX][nextY] == '#') {
                    direction = (direction + 1) % 4;

                    // Wenn ich in eine Koordinate laufe, an deren Stelle ich schon einmal in dieselbe Richtung gelaufen bin...
                } else if (map[nextX][nextY] == (char)(direction+'0')) {
                    obstructions++;
```

```
                    break; // ... dann kann ich die Endlosschleife abbrechen
                } else {
                    guardX = nextX;
                    guardY = nextY;

                    // Markiere in der Map, dass ich hier bereits in die
aktuelle Richtung gelaufen bin
                    map[guardX][guardY] = (char)(direction+'0');
                }

                // Wenn ich die Spielfläche gleich verlasse, dann kann ich
abbrechen
                if (guardX == 0 || guardX == width - 1 || guardY == 0 ||
guardY == height - 1) {
                    break;
                }
            }
            map[x][y] = '.';
        }
    }

    System.out.println(obstructions);
}
```