Aufgabe 2: Dreieckspuzzle

Team-ID: 00372

Team-Name: Der Teufel schreibt Java

Bearbeiter/-innen dieser Aufgabe:
Frederico Aberle

23. November 2020

Inhaltsverzeichnis

1 Lösungsidee

Die Lösungsidee ist die Brute-Force-Methode. Es sollen alle Möglichkeiten probiert werden, wie man die Puzzle auf dem 9er Dreieck platzieren kann. Dabei sollen auch alle Puzzleteile gedreht werden.

2 Umsetzung

Zunächst sollen alle Puzzleteile in einer Liste gespeichert werden und alle drei Ziffern eines Puzzleteils selbst auch in einer Liste (z.B. [[-1, -2, 1], [2, -1, -1], [-1, -2, 2], [-1, 3, 1], [2, -3, 3], [-1, 3, -2], [2, 2, -1], [-3, 2, -1], [-2, 1, -3]]). Dabei steht die erste Ziffer eines Puzzleteils für die erste Figur, die zweite Ziffer für die zweite Figur und die dritte Ziffer für die dritte Figur des Puzzleteils. Das erste Element der ganzen Puzzleteil-Liste steht dabei für die erste Karte (Puzzleteil), das zweite Element für die zweite Karte, das dritte Element für die dritte Karte, usw.. Wie man in Abbildung 1 sehen kann habe ich erste, zweite und dritte Karte in der Mitte definiert, da dies die Karten sind, die drei anliegende Kartenseiten haben. Karten 4, 5 und 6 haben nur zwei anliegende Kartenseiten und liegen deshalb zwischen den Karten 1, 2 und 3 und die Karten 7, 8 und 9 mit nur einer anliegenden Kartenseite befinden sich an den Ecken vom großen Dreieck.

def findSolution() Diese Funktion probiert nun alle Möglichkeiten aus, die Karten anzuordnen. Wenn eine Lösung gefunden wurde soll diese ausgegeben werden. Falls keine dieser Möglichkeiten eine Lösung ergibt soll ‚Es gibt keine Lösung' ausgegeben werden.

Zuerst werden in der Funktion drei for-Schleifen für die ersten drei Karten benutzt. Sie durchlaufen die Indexe der Liste, geht also von 0 bis 8. Dabei dürfen alle drei Karten nicht den gleichen Index haben, sonst würde man ja die Karten doppelt oder dreifach benutzen. Danach verwendet man wieder drei for-Schleifen, aber diesmal von 0 bis 2. Diese gibt an wie oft die Karte gedreht werden soll. Das heißt wird die Karte einmal gedreht, verschieben sich dementsprechend die Ziffern der Karte - per meiner Definition - auch um eins nach links. Dann werden Ziffer 2 zu Ziffer 1, Ziffer 3 zu Ziffer 2 und Ziffer 1 zu Ziffer 3 (d.h. [-1, -2, 1] wird zu [-2, 1, -1]). Analog, werden bei zweimaligem Drehen die Ziffern um zwei Stellen nach links verschoben (d.h. [-1, -2, 1] wird zu [1, -1, -2]).

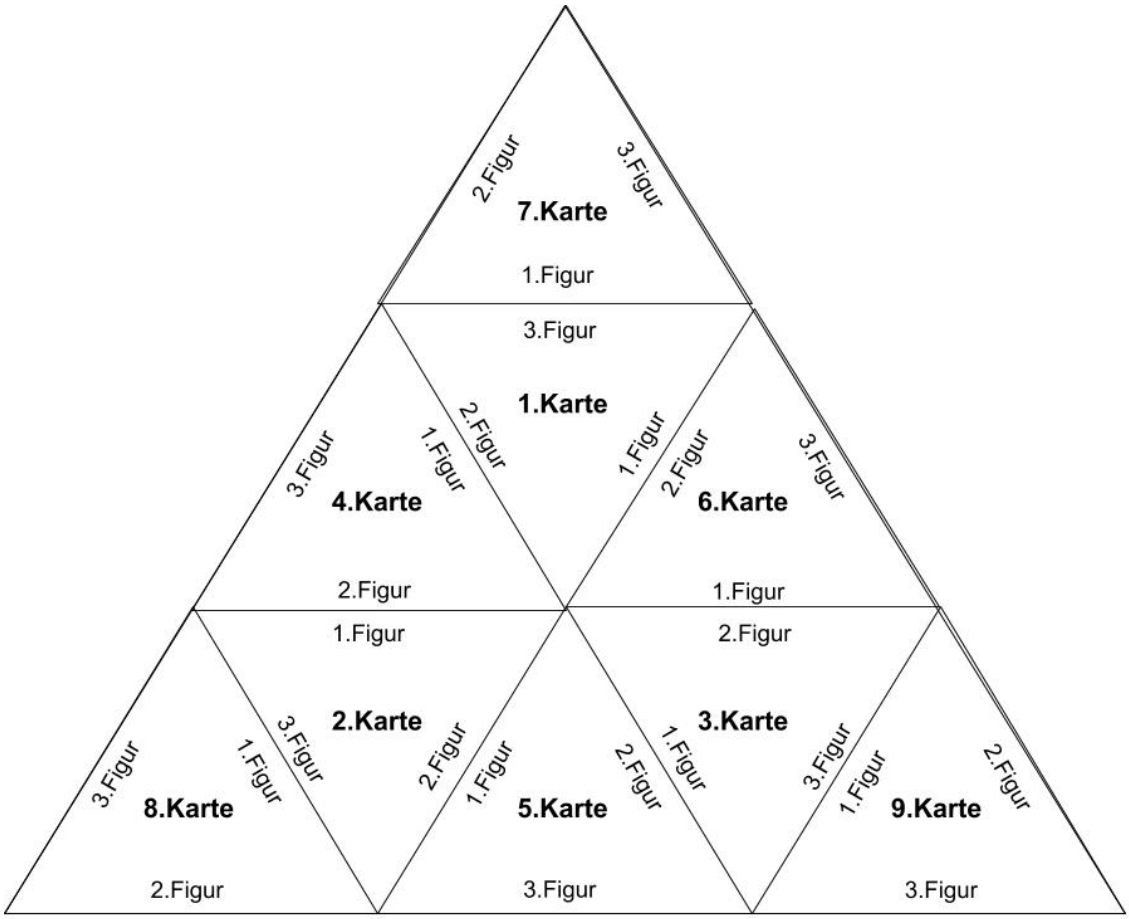

Abbildung 1: Dreieckpuzzle

Anschließend wird dieses Prinzip auch für die Karten 4, 5 und 6 angewendet. Wir verwenden für Karte 4 eine for-Schleife von 0 bis 8 für die Kartenauswahl (die Karte muss ungleich die Karten 1, 2 und 3 sein). Dann verwenden wir noch eine for-Schleife von 0 bis 2, die das Verschieben angibt. Die erste Ziffer ist dann entweder Ziffer 1, 2 oder 3 und die zweite Ziffer ist entweder Ziffer 2, 3 oder 1. Es sind nur die ersten beiden Ziffern notwendig, da wir diese zum Vergleichen brauchen (die dritte Ziffer grenzt an keine Karte an). Die Funktion soll nämlich weiterlaufen, wenn die 1. Ziffer der Karte 4 die Gegenzahl von der 2. Ziffer der Karte 1 ist und zusätzlich wenn die 2. Ziffer der Karte 4 die Gegenzahl von der 1. Ziffer der Karte 2 ist (siehe Abbildung 1). Nur dann sind die Figuren zueinander passend. Das wiederholt man auch für die Karten 5 und 6 mit dem Unterschied, dass die Ziffern 1 und 2 der Karte 5 mit den Ziffern 2 und 1 der Karten 2 und 3 passen soll und die Ziffern 1 und 2 der Karte 6 mit den Ziffern 2 und 1 der Karten 3 und 1 passen soll. Die Karten 7, 8 und 9 werden auch wieder iteriert von 0 bis 8 und aus der Liste ausgewählt und diese sind auch ungleich der Karten ihrer Vorgänger. Alle drei Karten werden wieder iteriert von 0 bis 2 nach ihren Drehmöglichkeiten. Am Ende wird jeweils bei jeder der Karten 7, 8 und 9 geprüft, ob ihre 1. Ziffer mit der 3. Ziffer der Karten 1, 2 und 3 passen.

Wenn alle Bedingungen der neun Karten erfüllt sind haben wir unsere Lösung gefunden. Wenn nicht, gibt es keine Lösung.

Lesen der Ausgabe: In der Ausgabe werden zunächst die ersten drei Karten ausgegeben gefolgt von ihren Anzahl an Drehungen. Dabei müssen Ziffern – wie oben beschrieben – x Drehungen nach links verschoben werden. Das bedeutet anschaulich am Beispiel des Dreiecks, dass die Kärtchen x mal gegen den Uhrzeigersinn gedreht werden. Danach folgen die nächsten drei Karten und ihre drei Drehungen und die letzen drei Karten und ihre drei Drehungen.

3 Beispiele

Die eigenen Beispiele beginnen ab puzzle4.txt

puzzle0.txt

Ausgabe:

([-1, -2, 1], [-2, 1, -3], [2, -3, 3], 0, 0, 0, [2, 2, -1], [-1, -2, 2], [-1, 3, 1], 0, 0, 1, [2, -1, -1], [-1, 3, -2], [-3, 2, -1], 1, 1, 0)

puzzle1.txt

Ausgabe:

([-1, -1, 3], [1, -2, -3], [3, -2, 2], 0, 0, 0, [1, -1, -2], [2, -3, -1], [1, -1, 2], 0, 0, 2, [-3, 3, -1], [-1, 3, 3], [-1, -2, 3], 0, 1, 1)

puzzle2.txt

Ausgabe:

([-2, -4, 1], [2, 3, 1], [3, 4, -1], 1, 1, 2, [-3, -2, -1], [-3, -1, 1], [-2, -3, 4], 2, 1, 1, [1, 2, -4], [-3, 1, -2], [-3, -2, -4], 1, 2, 2)

puzzle3.txt

Ausgabe:

([9, 8, -7], [2, 3, -4], [6, 5, -2], 1, 0, 2, [-2, 10, 7], [10, -3, 2], [-6, -8, 10], 2, 1, 0, [-9, 10, 10], [10, 10, 4], [10, -5, 10], 0, 2, 1)

puzzle4.txt

Eingabe:

10
9
10 10 1
9 8 -7
10 -5 10
-2 10 7
6 5 -2
2 3 -4
-6 -8 10
-9 10 10
10 -3 2

Ausgabe:

Es gibt keine Lösung

4 Quellcode

```
"""erstes,zweites und drittes Puzzle"""
for i in range(0,9):
        for j in range(0,9):
            for k in range(0,9):
                if i != j and i != k and j != k:
                    firstCard = cards[i]
                    secondCard = cards[j]
                    thirdCard = cards[k]
                    for l in range(0,3):
                        for m in range(0,3):
                            for n in range(0,3):
                                if l == 0:
                                    firstCardFirstFigure = firstCard[0]
                                    firstCardSecondFigure = firstCard[1]
                                    firstCardThirdFigure = firstCard[2]
                                elif l == 1:
                                    firstCardFirstFigure = firstCard[1]
                                    firstCardSecondFigure = firstCard[2]
                                    firstCardThirdFigure = firstCard[0]
                                elif l == 2:
                                    firstCardFirstFigure = firstCard[2]
                                    firstCardSecondFigure = firstCard[0]
                                    firstCardThirdFigure = firstCard[1]
                                if m == 0:
                                    secondCardFirstFigure = secondCard[0]
```

```
                                    secondCardSecondFigure = secondCard[1]
                                    secondCardThirdFigure = secondCard[2]
                                elif m == 1:
                                    secondCardFirstFigure = secondCard[1]
                                    secondCardSecondFigure = secondCard[2]
                                    secondCardThirdFigure = secondCard[0]
                                elif m == 2:
                                    secondCardFirstFigure = secondCard[2]
                                    secondCardSecondFigure = secondCard[0]
                                    secondCardThirdFigure = secondCard[1]
                                if n == 0:
                                    thirdCardFirstFigure = thirdCard[0]
                                    thirdCardSecondFigure = thirdCard[1]
                                    thirdCardThirdFigure = thirdCard[2]
                                elif n == 1:
                                    thirdCardFirstFigure = thirdCard[1]
                                    thirdCardSecondFigure = thirdCard[2]
                                    thirdCardThirdFigure = thirdCard[0]
                                elif n == 2:
                                    thirdCardFirstFigure = thirdCard[2]
                                    thirdCardSecondFigure = thirdCard[0]
                                    thirdCardThirdFigure = thirdCard[1]

"""viertes und fuenftes Puzzle"""
for p in range(0,9):
    fourthCard = cards[p]
    if p != i and p != j and p != k:
        for q in range(0,3):
            fourthCardFirstFigure = fourthCard[q]
            if q == 2:
                fourthCardSecondFigure = fourthCard[0]
            else:
                fourthCardSecondFigure = fourthCard[q+1]

            if fourthCardFirstFigure == firstCardSecondFigure*(-1)
                if fourthCardSecondFigure == secondCardFirstFigure*(-1):
                    for r in range(0,9):
                        fifthCard = cards[r]
                        if r != i and r != j and r != k and r != p:
                            for s in range(0,3):
                                fifthCardFirstFigure = fifthCard[s]
                                if s == 2:
                                    fifthCardSecondFigure = fifthCard[0]
                                else:
                                    fifthCardSecondFigure = fifthCard[s+1]

"""siebtes und achtes Puzzle"""
if seventhCardFirstFigure == firstCardThirdFigure*(-1):
    for c in range(0,9):
        eighthCard = cards[c]
        if c != i and c != j and c != k:
            if c != p and c != r and c != t:
                if c != a:
                    for d in range(0,3):
                        eighthCardFirstFigure = eighthCard[d]

                        if eighthCardFirstFigure == secondCardThirdFigure*(-1):
                            for e in range(0,9):
                                ninthCard = cards[e]
                                if e != i and e != j and e != k:
                                    if e != p and e != r and e != t:
                                        if e != a and e != c:
                                            for f in range(0,3):
                                                ninthCardFirstFigure = ninthCard[f]
```