Übungsblatt 2

Abgabe: 17.05.2024

Auf diesem Übungsblatt macht die Person die Implementierung, die in einem Telefonbuch zuletzt gelistet würde. Die andere Person macht die Tests.

Aufgabe 1 Insektenhotel

Auf einen Kalenderblatt der „Bunte Rätselwelt 2024“ findet sich das folgende Rätsel:

> Es summt und brummt im Insektenhotel. Belegte Nistplätze sind mit einem X markiert, noch freie mit einem O. Sie sind so auf die Kästchen zu verteilen, dass in jeder Zeile und jeder Spalte beides gleich oft vorkommt. Senkrecht und waagerecht dürfen nicht mehr als zwei belegte oder freie Plätze nebeneinanderliegen.

X							
	O	O					
			X				O
O							
		X	O			X	
X		X					
					X		
					X		

Dieses Rätsel soll nun mit Backtracking gelöst werden. Im Prinzip nicht nur dieses, sondern alle gleichartigen mit einer rechteckigen Form. Die Idee ist, dass eine Folge aus den Zeichen X und O gefunden werden soll, die der Reihe nach in die freien Kästchen des Rätsels eingefügt werden kann und somit das Rätsel löst. Das Erzeugen dieser Folge und damit das Lösen des Rätsels wird in Aufgabe 1.3 implementiert. In Aufgabe 1.1 wird das Ausfüllen des Rätsels umgesetzt und in Aufgabe 1.2 die Überprüfung, ob ein so ausgefülltes Rätsel den Regeln des Puzzles entspricht. Für alle drei Aufgaben gibt es bereits dokumentierte Signaturen in der Klasse *InsectHotel*.

Es ist sehr zu empfehlen, dass die Implementierungen von Aufgabe 1.1 und Aufgabe 1.2 ausführlich getestet werden, bevor mit Aufgabe 1.3 begonnen wird, da letztere von den ersten beiden abhängt.

Aufgabe 1.1 Ausfüllen (30 %)

Die Methode *fill* bekommt das eigentliche Rätsel in Form eines rechteckigen String-Arrays übergeben, in dem bereits einige Kästchen mit X oder O belegt sind. Die anderen sind leer (Leerzeichen). Zusätzlich bekommt die Methode eine Abfolge von X und O übergeben, mit der sie die freien Plätze des Rätsels ausfüllen muss. Das Ausfüllen passiert in einem neuen String-Array, d.h. die Parameter von *fill* werden nicht verändert. Das Ausfüllen passiert immer von oben links nach unten rechts (zeilenweise). Wenn ein Kästchen bereits im Rätsel belegt ist, wird sein Inhalt übernommen. Ist es hingegen frei, wird der nächste Eintrag aus der übergebenen Folge verwendet. Die Folge kann kürzer sein als die Anzahl der freien Plätze. Wenn es keine Einträge mehr in der

Folge gibt, werden ab dann immer die Kästchen aus dem Rätsel übernommen, d.h. auch leere Kästchen. *fill* gibt das (teil-)ausgefüllte Rätsel als Ergebnis zurück.

Aufgabe 1.2 Prüfen (40 %)

Die Methode *check* bekommt ein (teil-)ausgefülltes Rätsel übergeben und überprüft, ob bereits eine dieser Regeln verletzt ist:

- Es folgen horizontal und vertikal nicht mehr als zwei gleiche Einträge aufeinander. Noch leere Kästchen zählen dabei nicht mit.
- Es befinden sich horizontal und vertikal jeweils gleich viele X und O in einer Zeile bzw. Spalte. Dies gilt nur für vollständig gefüllte Zeilen bzw. Spalten. Es kann oft aber auch bei teilweise gefüllten Spalten bzw. Zeilen erkannt werden, dass die noch freien Kästchen nicht mehr so belegt werden können, dass gleich viele X und O vorhanden sind.

check gibt einen dieser drei Werte des Aufzählungstyps *Result* zurück:

INVALID: Die Belegung verletzt eine der Regeln.

VALID: Es wird keine Regel verletzt, aber das Rätsel ist noch nicht zu Ende ausgefüllt.

SOLVED: Es wird keine Regel verletzt und das Rätsel ist vollständig ausgefüllt.

Aufgabe 1.3 Lösen (30 %)

Durch die vorherigen zwei Aufgaben ist es nun möglich, eine Folge aus den Zeichen X und O in das Insektenhotel einzufügen und dann zu überprüfen, ob die resultierende Belegung erlaubt und vielleicht sogar die Lösung ist. In dieser Aufgabe wird nun die eigentliche Suche nach der Lösung in der Methode *solve* implementiert. Hierzu muss lediglich, analog zur Vorlesung, eine immer längere Folge erzeugt werden.[1] Für das Ende der Folge werden jeweils der Reihe nach die Werte X und O gewählt (die Reihenfolge ist egal) und das Ergebnis mit *check* überprüft:

INVALID: Die aktuelle Folge ist ungültig, also muss für das letzte Element der Folge der nächste Wert ausprobiert werden.

VALID: Die aktuelle Belegung ist gültig, also kann ein weiteres Element an die Folge angehängt werden, das durchprobiert wird.

SOLVED: Das Rätsel ist gelöst und wird als Ergebnis zurückgeliefert.

Immer, wenn das letzte Element der Folge bereits erfolglos X und O durchlaufen hat, muss Backtracking angewendet werden, d.h. es wird wieder entfernt und das Element davor „weitergezählt“. Entsteht beim Backtracking die leere Folge, gibt es keine Lösung. Dann gibt *solve null* statt einer Lösung zurück.

[1] Die Methode *fill* erwartet zwar einen *String*, es ist aber etwas einfacher, für das Verwalten der Folge die Klasse *StringBuilder* mit ihren Methoden *append*, *charAt* und *deleteCharAt* zu verwenden, und daraus für die Übergabe an *fill* mit *toString* wieder einen *String* zu machen.