Dr. Th. Letschert
Softwarekomponenten

THM TECHNISCHE HOCHSCHULE MITTELHESSEN
MNI Mathematik, Naturwissenschaften und Informatik

Aufgabenblatt 10

Aufgabe 1

Ein Irrgarten ist eine Fläche die teilweise frei und teilweise durch Wände aus botanischen Elementen versperrt ist. Es handelt sich also um ein erneuerbares, klimaneutrales, botanisches zweidimensionales Labyrinth. Der Begriff “Irrgarten” wird aber auch sehr oft im übertragenen Sinn gebraucht und ist dann ein Synonym zu “Labyrinth”.

Ein Irrgarten kann durch einen Graph dargestellt werde. Jede Stelle im Irrgarten, an der man die Wahl hat, in verschiedene Richtungen weiter zu gehen, entspricht dabei einem Knoten. Die Kanten sind die dort möglichen Wege. Der Eingang und der Ausgang sind ausgezeichnete Knoten. Der Irrgarten in Abb.1

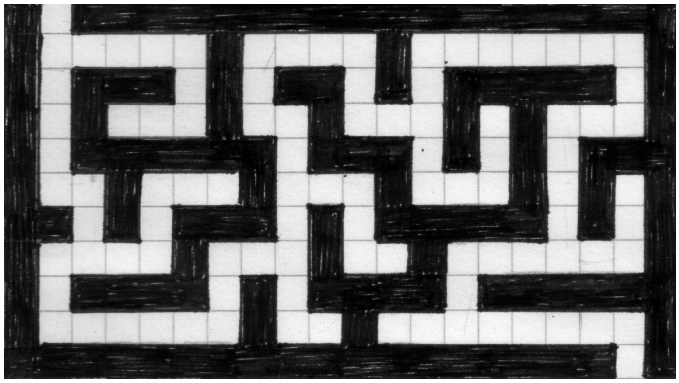

Abbildung 1: Ein Irrgarten

kann beispielsweise wie in Abb. 2 als Matrix mit ‘Verweigungspositionen”

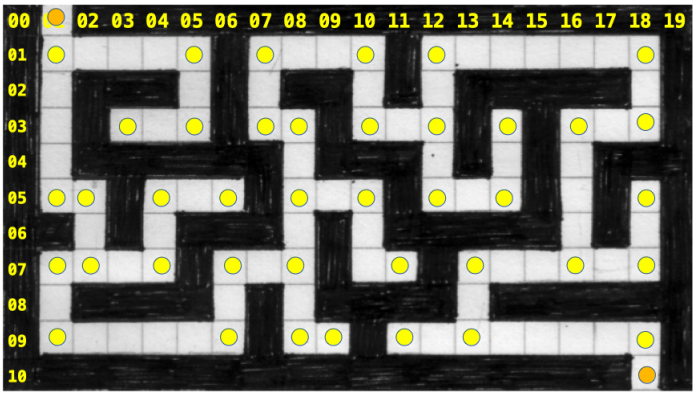

Abbildung 2: Der Irrgarten mit Verzweigunfspositionen

gesehen werden und diese dann als Graph mit den Verzweigungspositionen

$(0,1)$
$(1,1),\ (1,5),\ (1,7),\ (1,10),\ (1,12),\ (1,18)$
$(3,3),\ (3,5),\ (3,7),\ (3,8),\ )(3,10),\ (3,12),\ (3,14),\ (3,16),\ (3,18)$
$(5,1),\ (5,2),\ (5,4),\ (5,6),\ (5,8),\ (5,10),\ (5,12),\ (5,14),\ (5,18)$
$(7,1),\ (7,2),\ (7,4),\ (7,6),\ (7,8),\ (7,11),\ (7,13),\ (7,16),\ (7,18)$
$(9,1),\ (9,6),\ (9,7),\ (9,8),\ (9,11),\ (9,13),\ (9,18)$
$(10,18)$

als Knoten gesehen werden. Nummeriert man die Knoten von 0 bis 41, dann hat man deren Verbindungen:

$(0,1)$
$(1,2),\ (1,16),\ (2,8),\ (3,4),\ (3,9),\ (4,11),\ (5,12),\ (5,6),\ (6,15)$
$(7,8),\ (9,10),\ (10,20),\ (11,12),\ (12,22),\ (13,23),\ (14,32),\ (14,15)$
$(16,17),\ (17,26),\ (18,27),\ (18,19),\ (20,29),\ (20,21),\ (22,23),\ (24,33))$
$(25,34),\ (25,26),\ (26,27),\ (28,35),\ (28,29),\ (29,36),\ (31,39),\ (31,32),\ (32,33)$
$(34,35),\ (36,37),\ (38,39),\ (39,40),\ (40,41)$

als die Kanten des Graphs.

1. Definieren Sie eine **imperative** "Funktion" mit **Exception**-basiertem **Backtracking** die (nur genau) **einen** Weg durch ein Labyrinth sucht und verwenden Sie zur Suche eines Wegs von Knoten 0 (Position(0,1)) zu Knoten 41 (Position (10,18)) im Beispiel oben.

   Hinweis: Scala ist primär eine funktionale Sprache, der unbedachte Gebrauch von **return** kann zu Problemen führen. Vermeiden Sie es darum auch in imperativen Programmen mit **return** aus einer geschachtelten Kontrollstruktur heraus zu springen. **return** wird auf Exception-Handling abgebildet und kann mit dem selbst definierten Exception-Handling Ihres Algorithmus' kollidieren. Verwenden Sie statt **return** besser ein **throw**. (Oder Sie informieren sich an Hand der Scala-Dokumentation genauer zu diesem Thema.)

2. Transformieren Sie die imperative Lösung mit Hilfe von *Continuations* in eine funktionale Form.

3. Bringen Sie die funktionale Variante Ihrer Lösung in eine monadische Form.