

Aufgabe 1

$2 + 2 + 2 = 6T$

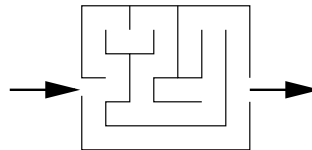
Ariadne ohne Faden

Eines Tages will die kluge Ariadne auf dem großen Markt des Nachbarortes Schuhe einkaufen gehen. Ganz geheuer ist ihr dabei nicht, denn es gehen Gerüchte darüber um, daß der Markt derart unübersichtlich sei, daß schon viele Leute in ihrem Kaufrausch den Ausgang nicht mehr gefunden haben sollen und auf ewig verschollen seien.

Dieses Schicksal will Ariadne auf keinen Fall teilen. Aber dennoch ist sie bereit, für ihre neuen Schuhe das Risiko einzugehen. Um die Gefahr so klein wie möglich zu halten, überlegt sie sich folgendes:

- Sie will sich alle besuchten Kreuzungen merken, damit sie nicht im Kreis läuft. Dies macht sie durch Nummerierung der Kreuzungen mit Kreide.
- Sie durchläuft den Markt einmal mit Breitensuche (von links nach rechts), um den nächstgelegenen Schuhstand zu finden.
- Sie durchläuft den Markt einmal mit Tiefensuche (von links nach rechts), um einen bestimmten, ihr sehr empfohlenen, Stand zu finden.

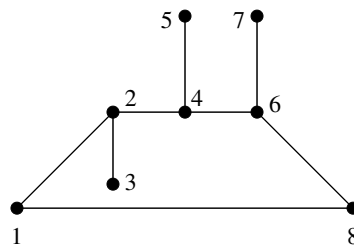
Was Ariadne nicht weiß, ist, daß es nur einen einzigen Schuhladen gibt, der direkt am anderen Ausgang des Marktes liegt. Der Markt hat folgende Gestalt:



- Geben Sie einen ungerichteten Graphen $G = (V, E)$ an, der die Gegebenheiten des Marktes widerspiegelt. Dabei sollen Pfade, für die es nur eine Möglichkeit gibt, als **eine** Kante dargestellt werden.
- Geben Sie einen möglichst einfachen abstrakten Datentyp an, mit dem Ariadne die Breitensuche realisieren kann. Welche Zustände nimmt diese Datenstruktur bei jedem von Ariadnes Schritten im Graphen G an?
- Geben Sie einen möglichst einfachen abstrakten Datentyp an, mit dem Ariadne die Tiefensuche realisieren kann. Auch hier sollen alle Zustände verdeutlicht werden.

Lösung zu Aufgabe 1

- $G := (\{1, \dots, 8\}, E)$, wobei E definiert wird durch



- Bei einer Breitensuche werden die Kinder des aktuellen Knotens immer erst ganz am Schluß expandiert. Dies läßt sich realisieren mit einer Schlange.

— 1 — 2,8 — 8,4,3 — 4,3,6

Da Ariadne nur den nächstgelegenen Schuhladen sucht, hört sie mit ihrer Suche auf, sobald Knoten 8 expandiert wurde.

- c) Bei einer Tiefensuche werden die Kinder des aktuellen Knotens als nächstes abgearbeitet. Deswegen ist der Datentyp eines Kellers besonders geeignet.



Da Ariadne nur den Schuhladen am Ausgang des Labyrinths sucht (Knoten 8), wird die Suche beendet, wenn der Knoten 8 zum ersten Mal expandiert wird.

Das mehrfache Vorkommen einzelner Knoten in der Schlange läßt sich dadurch erklären, daß ein Knoten erst dann mit Kreide markiert werden kann, wenn er bereits expandiert wurde (d.h. Ariadne kann natürlich nur an den Kreuzungen Kreidemarkierungen vornehmen, an denen sie bereits steht).

Aufgabe 2

4T

Such-Bäume

Implementieren Sie analog zur in der Vorlesung vorgestellten Linksrotation die Rechtsrotation. Verwenden sie dazu die Pseudo-Code Notation. Zur Erinnerung ist hier noch einmal die Definition der Linksrotation:

```

0  LEFT-ROTATE(T,x)
1  y ← rechts[x]
2  rechts[x] ← links[y]
3  if links[y] ≠ nil[T]
4      then p[links[y]] ← x
5  p[y] ← p[x]
6  if p[x] = nil[T]
7      then wurzel[T] ← y
8      else if x = links[p[x]]
9          then links[p[x]] ← y
10         else rechts[p[x]] ← y
11 links[y] ← x
12 p[x] ← y

```

Lösung zu Aufgabe 2

a) Rechtsrotation:

```
0 RIGHT-ROTATE(T,y)
1 x ← links[y]
2 links[y] ← rechts[x]
3 if rechts[x] ≠ nil[T]
4   then p[rechts[x]] ← y
5 p[x] ← p[y]
6 if p[y] = nil[T]
7   then wurzel[T] ← x
8   else if y = links[p[y]]
9         then links[p[y]] ← x
10        else rechts[p[y]] ← x
11 rechts[x] ← y
12 p[y] ← x
```

Aufgabe 3

15R

Happy Parsergenerator

Benutzen Sie den Parsergenerator `happy` um aus der Grammatik aus Aufgabe 3 von Übungsblatt 1 einen Parser für prädikatenlogische Formeln zu generieren.

Hinweise:

- Die Datei `Term.y` auf der Webseite definiert einen Parser für Terme. Ein Aufruf `happy Term.y` in der Kommandozeile generiert die Datei `Term.hs` mit der Parserfunktion `parseTerm`.
- Bei der Definition des Parsers wurden zwei Fehler in den Lösungshinweisen zu Aufgabe 3 von Übungsblatt 1 korrigiert.
- Die Symbole für Existenz- und Allquantor wurden als “A” und “E” festgelegt.

Lösung zu Aufgabe 3

Die Lösung ist in der Datei `Parse.y`.