Aufgabe 7 (trenne)

Schreiben Sie ein Programm

```
trenne(+Eingabeliste, -ZahlenListe, -NichtzahlenListe)
```

das eine Eingabeliste in eine Zahlen- und eine Nichtzahlenliste trennt. Um festzustellen, ob ein Term eine Zahl ist, benutzen Sie das Prädikat `number(X)` von Prolog. Verwenden Sie außerdem das Prädikat `append(X, Y, XY)`, welches zwei Listen `X` und `Y` zu einer Liste `XY` verkettet, z.B.

Frage: `?- append([a], [b,c], XY).`
Antwort: `XY = [a,b,c]`

a) Einfach geht es so, dass die Elemente in den Ausgabelisten in der umgekehrten Reihenfolge wie in der Eingabeliste erscheinen, also z.B.

Frage: `?- trenne([a,2,b,4,1,d,c], Zahlen, Nichtzahlen).`
Antwort: `Zahlen = [1,4,2], Nichtzahlen = [c,d,b,a])`

b) Können Sie es auch so, dass die Elemente in den Ausgabelisten in der gleichen Reihenfolge wie in der Eingabeliste erscheinen? Also, z.B.

Frage: `?- trenne([a,2,b,4,1,d,c],Zahlen,Nichtzahlen).`
Antwort: `Zahlen = [2,4,1], Nichtzahlen = [a,b,d,c])`

Aufgabe 8 (weg)

a) Schreiben Sie ein Programm

`weg(X, Y)`

das prüft, ob ein Weg von X nach Y existiert. Die Datenbasis lautet:

```
kante(1,2).
kante(1,3).
kante(2,4).
kante(2,5).
kante(3,4).
kante(4,7).
kante(5,6).
kante(6,7).
```

b) Schreiben Sie ein Programm

`weg(X, Y, N)`

das prüft, ob ein Weg von X nach Y mit der Länge N existiert. Für das Zuweisen von Werten hat Prolog das Prädikat `is`. Zum Beispiel liefert die Frage `?- N is 4 + 5.` als Antwort: `N = 11`

c) Schreiben Sie ein Programm

```
weg(X, Y, N, L)
```

das prüft, ob ein Weg von X nach Y mit der Länge N existiert und in einer Liste `L = [xn,...,x1]` die Punkte `xi` aus {1,2,...,7} ausgibt, die auf dem jeweiligen Weg von `x0` nach `xn` (in der Reihenfolge `x1,x2,...`) besucht wurden (nach dem Startpunkt `x0`).

Hinweis: Um eine Liste umzudrehen, hat Prolog das Prädikat `reverse/2`.

Beispiel:

```
Frage:    ?- weg(1,7,N,L).
Antwort:  N = 3
          L = [2, 4, 7];

          N = 4
          L = [2, 5, 6, 7]
```

Aufgabe 9 (trace)

Verwenden Sie den eingebauten Tracer, um festzustellen, in welcher Reihenfolge Prolog welche Aufrufe von `weg/4` und welche von `kante/2` macht, wenn Sie mit Ihrem Programm zu Aufgabe 8 die Frage

```
?- weg(2,7,N,L).
```

stellen. Genauer gesagt, machen Sie folgendes:

a) Schauen Sie mit `?- apropos(trace).` oder `?- help(trace).`, wie man Prolog sagt, von welchen Prädikaten es Informationen über
- Aufrufen (call) des Prädikats
- Benutzen der nächsten passenden Klausel des Prädikats (redo)
- Finden einer Lösung (exit)
- Scheitern einer Beweissuche für den Aufruf (fail)

ausgeben soll.

b) Sagen Sie Prolog, es soll
- für `weg/4` die Aufrufe, das Benutzen der nächsten Klausel, und das Finden einer Lösung,
- für `kante/2` nur die Aufrufe und die gefundenen Lösungen

melden.

Aufgabe 10

a) Schreibe ein Prolog-Praedikat,

```
element(+Atomarer Term,+Liste von atomaren Termen)
```

mit dem man feststellen kann, ob ein Term (hier: = Atomarer Term oder Liste) in einer Liste von atomaren Termen vorkommt.

Sie können annehmen, daß die Eingabeliste nur atomare Terme enthaelt.

b) Schreiben Sie ein Praedikat

```
difference(+Liste1,+Liste2,-Ergebnis),
```

das bei Eingabe zweier Listen von atomaren Termen, Liste1 und Liste2, im dritten Argument als Ergebnis eine Liste der Elemente ausgibt, die in Liste1 vorkommen, aber nicht in Liste2.

c) Schreibe ein Praedikat

```
insert(+Liste atomarer Terme, +atomarer Term,-erweiterte Liste)
```

das einen Term in eine Liste einfügt, wenn er darin noch nicht vorkommt.

Hinweis: Vergleiche den Term mit dem ersten Element der Liste ...

d) Schreibe ein Praedikat

```
union(+Liste1, +Liste2,-erweiterte Liste)
```

das die beiden Listen `Liste1` und `Liste2` zu einer erweiterten Liste vereinigt.