



Technische Universität Berlin

Software and Embedded Systems Engineering Group

Prof. Dr. Sabine Glesner

www.sese.tu-berlin.de

Sekr. TEL 12-4

Ernst-Reuter-Platz 7

10587 Berlin



Softwaretechnik und Programmierparadigmen WiSe 2022/2023

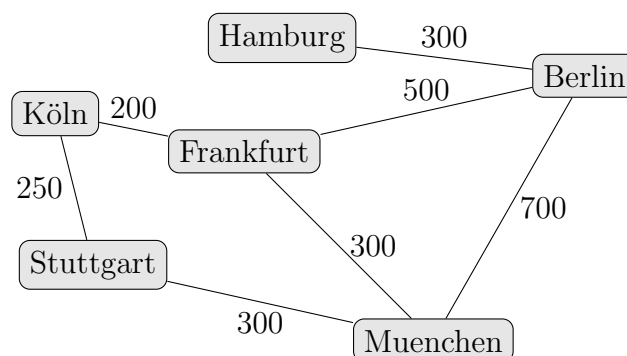
Prof. Dr. Sabine Glesner

Milko Monecke

Simon Schwan

Übungsblatt 06






Für die Bearbeitung dieses Blattes können die Ergebnisse aus Blatt 05 verwendet werden.
Es gilt weiterhin folgende Datenbasis:










Schlüssel:

- 📺 Ein ergänzendes Video wird zur Vor- oder Nachbereitung veröffentlicht.
- 🗣️ Wird im Tutorium besprochen.

Aufgabe 1: Rekursion

- a) Mit `connection(A,B)` soll nun herausgefunden werden, ob zwei Städte über beliebig viele Stopps miteinander verbunden sind. Welches Problem stellen die Zyklen auf unserer Karte dar? 
- b) `count(A,B)` gibt die Anzahl der Städte jeder Verbindung zwischen `A` und `B` aus. Tipp: ein "Hilfsprädikat" `countRec(A,B,Count)` kann hilfreich sein, wo `Count` die Anzahl der Städte auf dem Weg nach `A` ist. 
- c) `distance(A,B,Dist)` soll die Gesamtstrecke einer Verbindung ermitteln. Alternativ kann dieses mit einem Prädikat `distance(A,B,Dist,Erg)` implementiert werden, wo Zwischenergebnisse mit `Dist` "nach unten" gegeben werden und das Endergebnis in `Erg` erhalten wird. Probiert beide Optionen. 
- d) Das Prädikat `route(X,Y)` soll Routen mit beliebig vielen Zwischenstopps zwischen zwei Städten finden und alle Städte ausgeben können. Die Reihenfolge ist erst einmal egal. 
- e) Mit dem Prädikat `prime(X)` soll man prüfen können, ob `X` eine Primzahl ist. 

Aufgabe 2: Listen und Listenfunktionen

- a) Schreibt ein Prädikat `printList(L)`, mit dem eine Liste ausgegeben werden kann. 
- b) Schreibt ein Prädikat, mit dem eine Liste invertiert werden kann. 
- c) Das Prädikat `middle(L,Erg)` soll das mittlere Element einer Liste mit einer ungeraden Anzahl Elemente finden. Löst möglichst effizient. Was passiert, wenn man das Prädikat auf eine Liste mit gerader Anzahl Elemente anwendet? 
- d) Mit dem Prädikat `slice(LIn,IL,IR,LOut)` soll eine Sub-Liste von `LIn` mit den Elementen von Index `IL` bis `IR` (einschließlich, Indizes starten bei 0) in `LOut` erhalten werden. 
- e) Das Prädikat `route(X,Y,V)` soll nun die Verbindungen zwischen Städten in unserer Faktenbasis in der richtigen Reihenfolge ausgeben. Dafür müssen wir uns die besuchten Städte in der Liste `V` „merken“. Es soll außerdem sichergestellt werden, dass keine Stadt mehrfach besucht wird. Somit sollen keine unendlichen Routen mehr möglich sein.  
- f) `allRoutes(X,Y,ResultList)` soll alle möglichen Routen zwischen zwei Städten und ihre Längen in `ResultList` aufsammeln. Dazu muss die Funktion `Route` noch einmal so umgeschrieben werden, dass deren Ergebnisse nicht ausgegeben werden sondern in Variablen zur Verfügung stehen. Es können vordefinierte Prädikate verwendet werden. 

- g) Ziel der Übung ist es natürlich, die kürzeste Route zu finden. Erstellt das Prädikat `shortestRoute(X,Y,Shortest)`, mit dem die kürzeste Route und ihre Länge ermittelt werden kann.



Aufgabe 3: 4 Wikinger (Zusatzaufgabe)

Vier Wikinger sollen im Dunkeln in höchstens einer Stunde über eine Brücke gehen, haben dabei aber ein Problem: Die Brücke kann maximal nur zwei von ihnen tragen, außerdem haben sie nur eine Fackel dabei und es kann niemand ohne Fackel über die Brücke. Die Wikinger sind außerdem unterschiedlich schnell, sie brauchen zum Überqueren die folgenden Zeiten: (5min, 10min, 20min, 25min). Löst das Rätsel mithilfe von Prolog.