COMP 304 Assignment 5
(More Prolog)

Due Date: 7 October, 2016

Introduction

This project is worth 7.5% of your total course grade. It involves writing Prolog code. All your work must be submitted electronically. Prolog code files must be plain text files with a `.pl` extension. Any other material (which includes any additional descriptions of the way you chose to solve any of the problems) should be in `.pdf` or `.txt` format.

All your code must be properly commented. Use comments extensively to communicate the purpose of each predicate, its variables and clauses, and what the overall design of your program is. There will be a severe marking penalty for programs without comments.

Supply sample queries that demonstrate the functionality of your programs as requested in the following questions. These may derive from queries you have used in your own testing but make sure that your sample queries are not too high in numbers, yet still manage to demonstrate all functionality.

1 North Island Journey Planner (50%)

Table 1 describes the North Island's major road network and the distances between towns.

1.1 Road Database

Define a predicate `road/3` to capture this information. You will use this predicate to help plan journeys around the North Island.

One condition which we demand of any journey is that no town is visited twice; every station of a journey should provide a new town to enjoy.

From	To	km
Wellington	Palmerston North	143
Palmerston North	Wanganui	74
Palmerston North	Napier	178
Palmerston North	Taupo	259
Wanganui	Taupo	231
Wanganui	New Plymouth	163
Wanganui	Napier	252
Napier	Taupo	147
Napier	Gisborne	215
New Plymouth	Hamilton	242
New Plymouth	Taupo	289
Taupo	Hamilton	153
Taupo	Rotorua	82
Taupo	Gisborne	334
Gisborne	Rotorua	291
Rotorua	Hamilton	109
Hamilton	Auckland	126

Table 1: Connection data

1.2 Route Planning

Write a predicate `route/3` to plan routes: `route(Start, Finish, Visits)` should succeed if there is a route from `Start` to `Finish` visiting the towns in list `Visits`.

1.3 Route Planning With Distances

Write a predicate `route/4` to plan routes: `route(Start, Finish, Visits, Distance)` should succeed if there is a route from `Start` to `Finish`, visiting the towns in list `Visits`, which is `Distance` long.

1.4 Finding All Routes

There is a predefined Prolog predicate `findall/3` which you can use to find all the solutions to a goal. The first argument to `findall` is a Prolog term, the second argument is a Prolog goal and the third argument is a list. The list becomes bound to the list of instances of the term for which the goal succeeds. Using `findall` only makes sense if there are variables shared between the term and the goal. Often the term is just a variable, but it can be any term.

For example:

```
?- findall((X,Y), append(X, Y, [1,2]), Results).

Results = [ ([], [1, 2]), ([1], [2]), ([1, 2], [])].
```

Write a predicate `choice/3` to help plan routes: Predicate `choice(Start, Finish, RoutesAndDistances)` should produce a list of all the routes (including their distances) between `Start` and `Finish`.

1.4.1 Finding All Routes Including Towns

Write a predicate `via/4` to help planning routes: `via(Start, Finish, Via, RoutesAndDistances)` should produce a list of all the routes (including their distances) between `Start` and `Finish` which visit towns within `Via`.

1.4.2 Finding All Routes Avoiding Towns

Write a predicate `avoiding/4` to help plan routes: `avoiding(Start, Finish, Avoiding, RoutesAndDistances)` should produce a list of all the routes (including their distances) between `Start` and `Finish` which do not visit towns within `Avoiding`.

1.4.3 Testing

Give Prolog queries to show that `route/4`, `choice/3`, `via/4` and `avoiding/4` work properly.