Assignment 4

CS|E 141 & Inf4mtx 101: Programming Languages

Question 1: Train Travel

We have the following knowledge base:

```
nonStopTrain(sandiego,oceanside).
nonStopTrain(lasvegas,sandiego).
nonStopTrain(sanfrancisco,bakersfield).
nonStopTrain(bakersfield,sandiego).
nonStopTrain(oceanside,losangeles).
nonStopTrain(portland,sanfrancisco).
nonStopTrain(seattle,portland).
```

This knowledge base holds facts about towns it is possible to travel between by taking a *non-stop* train. We can travel further by `chaining together' non-stop train journeys.

Write a recursive predicate `routeTrip/3` that tells us whether we can travel by train between two towns, and the route taken.

For example, when given the query

```
?- routeTrip(seattle,oceanside, Trip).
     Trip = [seattle, portland, sanfrancisco, bakersfiled,
sandiego, oceanside] .
```

Also, encode a relationship to express that travel can happen in both directions: A to B, and B to A.

```
?- canTravel(oceanside,seattle).
     `yes'.
```

Question 2: The Seating Chart

Problem Description

As the host of a party, you are required to generate a seating chart for your guests. There will be 8-guests seated at a circular table. There are 4-male, 4-female guests and each guest has a unique set of hobbies. You decide that the following constraints must be met when seating guests:

1) Adjacent seating partners must be of the opposite gender
2) Adjacent seating partners must share at least one hobby

You are to write the predicate, `seatingChart(X)`, where `X` contains the names of the persons assigned to seats 1-8.

The guest information is as follows:

```
% Name  Gender  Hobbies
```

```
%               Hobby 1 Hobby 2 Hobby 3
% sue   f       yoga
% jim   m       chess
% tom   m       run     yoga
% joe   m       chess   run
% cami  f       yoga    chess   run
% bob   m        run    yoga
% fay   f       yoga    run     chess
% beth  f       chess   run
```

Implementation

Using SWI-Prolog, create one knowledge base for *all three* questions named assignment4.pl
For full credit, all solutions must be free of errors and warnings.