CSE341 – Programming Languages (Fall 2017)

Homework #2

Handed out: 10:00am Monday November 27, 2017.

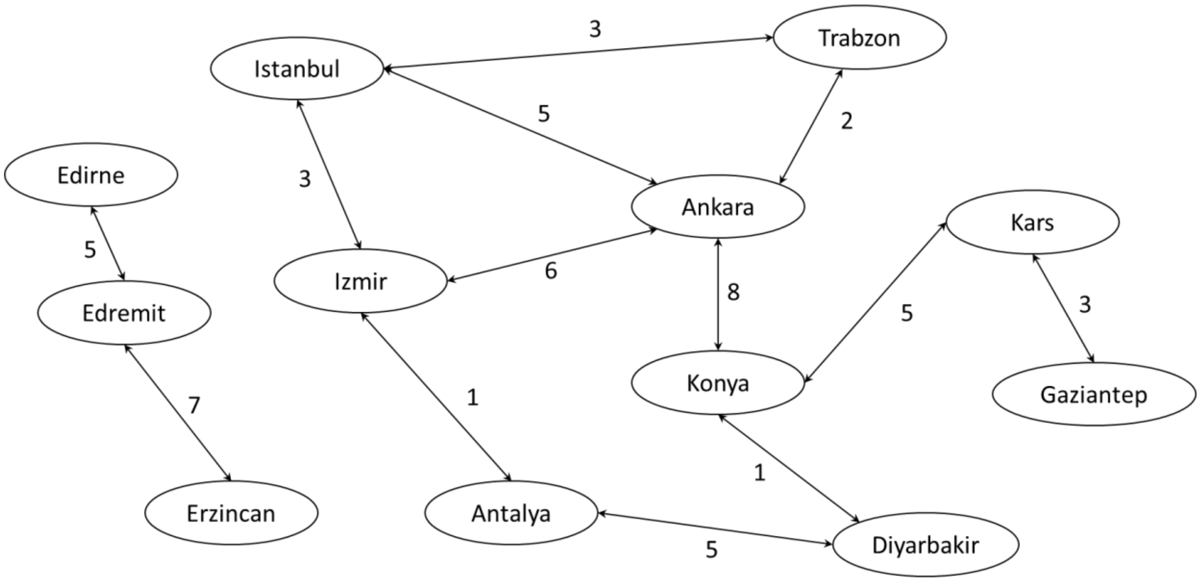
Due: 11:55pm Wednesday December 15, 2017.

Hand-in Policy: Source code and documentation should be submitted online as a single .zip or .rar file with naming convention studentid_lastname_firstname_h2.zip via Moodle by the submission deadline. No late submissions will be accepted.

Collaboration Policy: No collaboration is permitted. Any cheating (copying someone elses work in any form) will result in a grade of -100 for the first offense and -200 for the subsequent attempts.

Grading: Each homework will be graded on the scale 100. Unless otherwise noted, the questions/parts will be weighed equal.

Part 1. In the graph below you see the possible flights between some of the cities in Turkey. Write the predicate "route(X,Y,C) – a route between X and Y exists with cost C" that checks if there is a route between any given two cities.



Your program should have all the facts and predicates/rules. See the following:

```
% knowledge base

…

flight(istanbul,izmir,3).  % fact: Istanbul and Izmir has a flight with cost 3.

…

% rules

…

route(X,Y,C) :- flight(X,Y,C). % a predicate indicating there exist a route between
```

```
                                    % X and Y if there is flight between X and Y with cost
                                    % C.
     …
```

A single query to complete your program should check if there is a direct route between two given cities. Alternatively, it can list all the connected cities for a given city. See the following:

```
?- route(edirne,X,C).
X = erzincan, C = 12;
X = edremit, C = 5 ;
```

Part 2. Continuing with the previous problem, you are asked to write a program that checks if a route exists between two cities and if so provides the cheapest route.

A single query to complete your program should check if there is a direct route between two given cities and the shortest distance between them. See the following:

```
?- croute(edremit,erzincan,X).
X = 7 ;
```

Part 3. You are given the following database about a conference. There are sessions at the conference all day long. There are of course attendees enrolled for these sessions. Some of the sessions, rooms and enrollments are as follows.

Sessions (each lasts 2 hours)		
Name	Start Time	Room
A	10	101
B	12	104
C	11	102
D	16	103
E	17	103

Enrollment	
Attandee	Session
1	A
1	B
2	A
3	B
4	C
5	D
6	D
6	A

Write the predicates "when(X,Y) – time of the session X is Y", "where(X,Y) – place of the session X is Y", and "(X,Y) – X is enrolled in session Y".

3.0. Add at least 5 more attandees each enrolled to two or more sessions. Add one two more sessions (no new rooms should be added).

3.1. Define/write a predicate "schedule(S,P,T)" associates an attandee to a place and time of session.

3.2. Define/write another predicate "usage(P,T)" that gives the usage times of a room.

3.3. Define/write another predicate "conflict(X,Y)" that gives true if sessions X and Y conflict due to room or time.

3.4. Define/write another predicate “meet(X,Y)” that gives true if attandees X and Y are present in the same room at the same time.

Part 4. Write the following predicates operating on lists.

4.1. Define a Prolog predicate “union(L,S,U)” that finds the union of all the elements of L and S. The result will be bound to U.

4.2. Define a Prolog predicate “intersect(L1,L2,I)” that finds the intersection of L1 and L2 binding the results to I.

4.3. Define a Prolog predicate “flatten(L,F)” that flattens the list of lists (that elements of L can be list of lists, e.g. nested lists are possible) in L binding results to F.