Recursion: Work with your partner(s) to understand and write recursive Prolog relations.

Follow the instructions:

1. Login to phoenix. Download the ch4_1.pl file from goucherLearn. This file contains information about airline flights between cities.

2. We want to write a relation which tells us if we can "getTo" one city from another from the available flights. Clearly, we can get to city Y from city X if there is a direct flight from X to Y:

   ```
   getTo(X,Y) :- flight(X,Y).
   ```

   But it is also the case that we can get to Y if we can fly directly to another city Z and it is possible to get to Y from Z. This is *recursive* because we are using getTo within the definition of itself but it all makes sense.

   Open up the ch4_1.pl file and type out the recursive case for the getTo relation. Make sure it appears *after* the case for the direct flight. Save your changes, load the file in Prolog, and try it out with the queries:

   ```
   getTo(baltimore, minneapolis).

   getTo(minneapolis, miami).
   ```

3. What happens if we reverse the two cases. Try this out by moving the recursive case *before* the direct flight case. Try it out. Why do you think this happened?

4. Download the file ch4_2.pl from goucherLearn. Write (and try out) the following relations:

   a. sibling
      X is a sibling of Y if X and Y are two different people who share a parent in common.

   b. descendent
      X is a descendent of Y if X is either a child of Y or (recursively) of someone who is a descendent of y.

   c. cousin
      X is a cousin of Y if some parent of X and some parent of y are either siblings or (recursively) cousins.