Due at the end of Week 6: Friday, October 21, 11:55 pm. This is an individual assignment. Each student submits: assn1.pdf, spiders.lp, spidershortlegs.lp.
assn1.pdf contains your answers to the questions, including any relevant bits of code.
spiders.lp and spidershortlegs.lp contains the entirety of your finished program (it should not include sample data such as declaration of a specific graph through vertex/1 and edge/2, but it should include a line that prints out the desired output).

This assignment is graded out of 10 and contributes 10% to the course grade.

Question 1 *Spanning spiders*

A spanning tree of a graph is a subgraph that is a tree and contains all the vertices of the original graph. A spider is a graph with at most one vertex whose degree is 3 or more, this vertex is called the *center* of the spider. In this exercise, we are interested in creating a program to find *spanning spiders* of input graphs: for any graph, we want to identify a subgraph that is a spanning tree and also a spider. In a spanning spider, we call edges including the subgraph the *leg edges*.

This exercise is inspired from one of the questions in the 2003 Prolog Programming Contest.[1]

The goal is to write an Answer Set Program such that given a graph as input, every stable model corresponds to a distinct spanning spider. Not every graph has a spanning spider, so if an input graph does not admit any spanning spider, the corresponding program should be unsatisfiable.

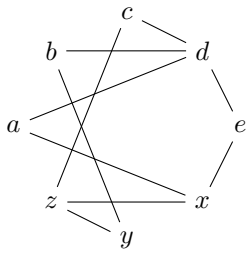

Figure 1: Example graph

Input format An input file contains predicate instances of `vertex/1` indicating the vertices of the graph and predicate instances of `edge/2` indicating the edges of the graph.

Output format Your program should compute a model containing a predicate `leg/2` indicating which edges are included in the spanning spider, as well as a predicate `center/1` indicating which vertex is the center of the spider.

```
vertex(a).
vertex(b).
vertex(c).
vertex(d).
vertex(e).
vertex(x).
vertex(y).
vertex(z).
edge(a,d).
edge(a,x).
edge(b,d).
edge(b,y).
edge(c,d).
```

[1] https://people.cs.kuleuven.be/~bart.demoen/PrologProgrammingContests/Contest2003.html

```
edge(c,z).
edge(d,e).
edge(e,x).
edge(x,z).
edge(y,z).
```

1.1 Does the graph in Figure 1 have any spanning spider? If yes, indicate the center vertex and the list of edges included in the spanning spider.

1.2 Provide ASP rules that define the `center/1` predicate and ensure that in any model, exactly one vertex is selected as the center.

1.3 Provide a generator for the `leg/2` predicate.

1.4 One property required of spanning spiders is that every vertex should be reachable from the center through leg edges. Introduce a derived predicate `reachable/1` that ranges over vertex names and is true when the corresponding vertex is reachable from the center through leg edges. Use this predicate to define a constraint that ensures every vertex of the original graph is reachable from the center through leg edges.

1.5 The reachability of every vertex from the center is a required property of spanning spiders, but it is not sufficient and we need to ensure other constraints are satisfied. Describe all the other constraints that need to be satisfied and write ASP rules to enforce these constraints.

1.6 Based on your answers to the above questions, write an ASP program `spiders.lp` that takes an input graph and outputs all the distinct spanning spiders of the graph. How many distinct spanning spiders does the graph in Figure 1 have?

1.7 Spanning spiders with short legs. Write an ASP program `spidershortlegs.lp` that outputs a spanning spider with the shortest longest leg.