Rule-based Programming

Assignment #5

Excercise 1 (Reflexive-Transitive Closure). Implement and experiment with the following program to calculate the transitive closure of a directed graph, where the `e/2` represents edges and `p/2` represents paths:

```
e(X,Y) ==> p(X,Y).
e(X,Y), p(Y,Z) ==> p(X,Z).
```

Modify the program such that it calculates the reflexive-transitive closure of the graph and make sure that it terminates even for cyclic graphs. Test your modification with the input `e(1,2), e(2,1)`.

Excercise 2 (Shortest Paths I). Instead of a binary now use a ternary path constraint `p/3` where `p(X,Y,N)` represents the fact that a path of length `N` exists from `X` to `Y`. Modify your program from Ex. 1 such that it calculates shortest paths in a graph.

Excercise 3 (Shortest Paths II). Modify your program from Ex. 2 such that it calculates only paths starting at a specific source node, represented as a unary constraint `source/1`.

Excercise 4 (Shortest Paths III). Now use weighted edges, represented by a ternary edge constraint `e/3` where `e(X,Y,N)` represents the fact that an edge of weight `N` exists from `X` to `Y`. Modify your program from Ex. 3 such that it calculates shortest paths in a weighted graph starting from a specific source node.

Excercise 5 (Shortest Paths IV). Modify your program from Ex. 4 such that it keeps a representation of each shortest path starting from the source node. Use a quarternary constraint `p/4` where `p(X,Y,N,P)` represents a path of length `N` from vertex `X` to vertex `Y` and P is a list of visited vertices.