Consider the Prolog knowledge base in basegraph.pl. The knowledge base consists of two parts.

The first part encodes a weighted undirected graph using edge/3 facts, where edge(v, u, c) indicates that nodes v and u are connected through an edge whose associated cost is the positive, real number c. In particular, basegraph.pl encodes one of the following graphs:

Note that basegraph.pl can only encode a single graph at a time, but the first part of the knowledge base can be changed so as to encode a different graph. For example, basegraph.pl initially encodes Graph 1, while the encoding for Graph 2 is commented in the code. You can un-comment and comment Graph 2 and Graph 1 encodings, respectively, and then re-compile it in your SWI-PL to make basegraph.pl encodes Graph 2 instead.

The second part of the knowledge base is instead fixed, and encodes two useful additional predicates:

- A predicate connection(N1, N2, C) that is true whenever N1 and N2 are two different nodes from the graph, such that there is an edge between them with cost C. Notice that this predicate is symmetric, i.e., connection(N1, N2, C) is true if and only if connection(N2, N1, C) is true.

- A predicate graph_nodes(L) that is true when L is a list containing the set of all nodes in the graph.
By exploiting these two predicates, you have to:

1. Write a predicate \texttt{complete} that is true if the graph encoded in the knowledge base is \textit{complete}, that is, when every pair of distinct nodes in the graph is connected by a unique edge. In the above examples, Graph 2 is complete whereas Graph 1 is not.

2. Write a predicate \texttt{path(S,T,P,L)} that is true if \texttt{P} is a path connecting \texttt{S} and \texttt{T} with total length \texttt{L}. For example, by asking all solutions of the query \texttt{?- path(s,t,P,L)}, we should obtain all paths connecting \texttt{s} with \texttt{t} (with their respective lengths).

3. Write a predicate \texttt{spath(S,T,P,L)} that is true if \texttt{P} is (one of) the shortest path(s) connecting \texttt{S} and \texttt{T}, with total length \texttt{L}. For example, by querying the knowledge base with \texttt{spath(s,t,P,L)}, we should get the following:

    \begin{verbatim}
    ?- spath(s,t,P,L).
    P = [s, n3, n5, t],
    L = 8.7.
    \end{verbatim}

4. Write a predicate \texttt{connected} that is true if the graph encoded in the knowledge base is \textit{connected}, that is, guarantees the existence of a path between every pair of vertices.

5. Write a predicate \texttt{hcycle(P,L)} that is true if \texttt{P} is a Hamiltonian cycle in the graph with total length \texttt{L}. A Hamiltonian cycle is a path in the graph that visits each vertex exactly once, and forms a cycle. For example, \texttt{[s, n2, n4, n5, t, n1, n3]} is a Hamiltonian cycle of Graph 1 with total length of 24.5.

6. Write a predicate \texttt{shcycle(P,L)} that is true if \texttt{P} is a (one of) the shortest Hamiltonian cycle with total length \texttt{L}. For example, by querying the knowledge base of Graph 1 with \texttt{shcycle(P,L)}, we should get the following:

    \begin{verbatim}
    ?- shcycle(P,L).
    P = [s, n2, n4, n5, t, n1, n3],
    L = 24.5.
    \end{verbatim}