CS_125 Logic Programming – Exercises 1 (not assessed)

Question 1. Define a predicate `last/2` computing the last member of a nonempty list.

Question 2. Define a predicate `different/3` such that for a list `L` without repetitions `different(X,Y,L)` means that `X,Y` are different members of `L`.

Hint: Use the predicates `select/3` and `member/2`.

Question 3. Referring to coursework 1, define predicates `sibling/2` and `aunt/2`.

Question 4. Define a predicate `list_of_lists/1` testing whether an object is a list of lists.

Question 5. Define a predicate computing for a given list of lists the concatenation of its members.

Question 6. Define a predicate `equal_sets/2` testing whether two lists represent the same set. For example, the lists `[a,b,a,c]` and `[b,c,c,c,a]` represent the same set $\{a, b, c\}$.

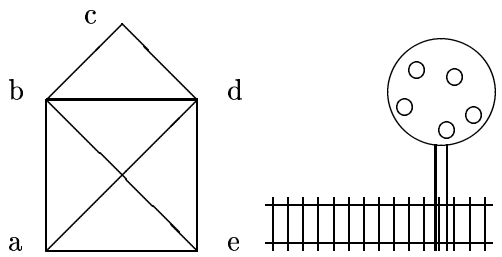
In the following we represent an undirected graph by a repetition free list of its edges. An edge between vertices `V` and `W` shall be represented by the term `edge(V,W)`. Since we regard graphs as non directed we consider `edge(V,W)` and `edge(W,V)` as equal edges.

Note that a single graph usually has many different representations. For example the graph given by a triangle with vertices `a,b,c` has the representation `[edge(a,b),edge(a,c),edge(b,c)]`, but also `[edge(b,a),edge(b,c),edge(c,a)]`, and many more.

Question 7. Define a predicate `equal_edge/2` testing whether two edges are equal.

Question 8. Write a program for drawing a graph, using a continuous line, without taking out the pencil from the surface of the paper, and without drawing a line twice.

Example:



The graph above, with vertices a,b,c,d,e, is given by the list of edges

```
[edge(a,b),edge(a,d),edge(a,e),edge(b,c),edge(b,d),edge(b,e), edge(c,d),edge(d,e)]
```

The drawing of this graph should be a list of vertices `[V1,V2,...,V9]`, where `V1,...,V9` are vertices, describing the course of the pencil (there are 8 edges, therefore the pencil must visit 9 vertices).

Use the following idea for a recursive definition of the drawing program: A list `[V1,V2|D]` is a drawing of a graph `G` if there is an edge `E` in `G` such that `E` is the same edge as `edge(V1,V2)` and `[V2|D]` is a drawing of the graph `G1`, where `G1` is obtained from `G` by removing `E`.

To start the recursion it is convenient to accept any singleton list as the drawing of an empty graph.

Find all drawings of the graph above starting at vertex `a`.

Are there drawings starting at vertex b ?

In the following we represent an English-French dictionary by a list of entries

```
[entry(one,un),entry(two,deux),entry(three,trois),...]
```

Question 9. Define a predicate `translate_word/2` translating an English word into French.

Question 10. Define a predicate `translate_sentence/2` translating an English sentence, given by a list of words, into French (of course the quality of this translation will be very poor).

Question 11. Define a predicate `lh/2` computing the length of a list as a numeral `0,s(0),s(s(0)),...`.

Question 12. Define a predicate `head_tails/3` that splits a list of nonempty lists into the list of heads and tails of its member. For example the query

```
?- head_tails([[a,b,c],[d], [e,f,g]],Heads,Tails).
```

should return

```
Heads = [a,d,e]
Tails = [[b,c],[],[f,g]]
```

Question 13. Define a predicate `rectangle/1` testing whether an object is a rectangle, that is, a list of lists of equal lengths.

Hint: The predicate `rectangle/2` can be defined either using the predicate `lh`, or, even better, using the following idea. A list of lists is a rectangle iff when repeatedly chopping off simultaneously the heads of the lists all lists become empty at the same time. This means that `rectangle/1` can be defined recursively using `head_tails/3`.

Question 14. Define a predicate `square/1` testing whether an object is a square.

Question 15. Define a predicate `matrix/2` testing whether an object is a matrix, that is, a rectangle of non-degenerate dimensions. Hence a matrix has at least one row and at least one column.

Question 16. Define a program for transposing matrices. For example the transpose of the matrix

```
[[a,b,c],
 [d,e,f]]
```

is the matrix

```
[[a,d],
 [b,e],
 [c,f]]
```

(rows become columns).