TDDD08 — Tutorial 5

1. Write a logic program defining a predicate `palindrome`/1 which is true if the argument is a list which is a palindrome.

A palindrome is a list which is the same after reversal, i.e. $[e_1, \ldots, e_n]$ where $n \geq 0$ and $e_{1+i} = e_{n-i}$ for $i = 0, \ldots, n-1$.

2. Write a logic program defining a predicate `thi/2` which is true when its arguments are lists, and the second list contains every third element of the first one, starting from the second element. For instance, `thi`$([a, b, c, d, e], [b, e])$ should hold.

3. Consider binary ordered trees, in which each leaf contains a value, which may be an arbitrary term (and non-leaf nodes do not contain values). Design a representation of such trees as terms.

Write a program checking that a tree is the symmetric image of another one.

(A tree t is the symmetric image of a tree s iff 1. both consist of the same single leaf, or 2a. the left subtree of the root of t is the symmetric image of the right subtree of the root of s, and 2b. the right subtree of the root of t is the symmetric image of the left subtree of the root of s.)

4. Consider the following Prolog program.

```
x([], [], []).
x([X|Xs], [X|Ys], Zs ) :- x(Xs, Zs, Ys).
```

Describe the intended purpose of this program and explain the relationship between its arguments. For example, what is the result of the query `x([a,b,c,d,e], Ys, Zs)`?

5. Assume that directed graphs are represented as lists of pairs, one pair for each node of graph. Each pair is of the form $t - [t_1, \ldots, t_n]$, where t is a node of the graph and $(t, t_1), \ldots (t, t_n)$ are the arcs coming out of t. For instance, we could have the following graph: $[a$ - $[b, c], b$ - $[c], c$ - $[d, e], d$ - $[f], e$ - $[f, g], f$ - $[g], g$ - $[e]\,]$.

Write a Prolog program with a predicate *three_cycle*/1 finding whether in a given graph there exists a cycle of length 3 (like the cycle e, f, g, e in the example graph).

6. Write a logic program defining predicates $el/1$ and $ol/1$; the first is true when its argument is a list of even length, the other – if it is a list of odd length. Your program should not use Prolog arithmetic, constraints, or negation.