CvP - Programming Assignment 3

- Deadline: November 23, at the start of the werkcollege.
- Submit your solution electronically via liacscvp2018@gmail.com
- Clearly state your name and student number your solution file.
- Put your progrm inside a .pl file and hand in this file, together with a written report (in .txt of .pdf) in which you explain your programs.
- For this assignment, we use the SWI-Prolog implementation of the Prolog language (http://www.swi-prolog.org/).
- For good tutorials check: http://www.learnprolognow.org/
- The Prolog interpreter has been compiled for Linux/x86 systems, and is available on the LIACS net- work at /home/csalp/bin.linux/prolog/pl. Versions for other systems are available for download at the aforementioned web-page. You can run the interpreter as follows:

  ```
  /home/csalp/bin.linux/prolog/pl
  ```

  or run the interpreter for a given file:

  ```
  /home/csalp/bin.linux/prolog/pl -f myprog.pl
  ```

  where `myprog.pl` is your Prolog program. Note: when using the first version, type `consult(myprog.pl).` or `[myprog].` at the beginning and every time you change the program.
- Use capital letters when using variables: use Name instead of name.
- When using functions, do not put a space between the function name and the first bracket: `f(...)` instead of `f (...)`.
- Put a period `.` at the end of every command, or they will not be executed.
- To use the implication symbol ($\leftarrow$) in SWI-Prolog, you have to use `:-`, i.e., for bird(X) $\leftarrow$ lays eggs(X) $\wedge$ has wings(X) we write:

  ```
  bird(X) :- lays_eggs(X), has_wings(X).
  ```

- You can exit SWI-Prolog by entering CTRL-C followed by an e.

Question 1 Consider binary trees whose nodes are labelled with natural numbers. Use the term `void` to denote the empty tree, and the term `tree(x, left, right)` to denote the tree with root `x`, left subtree `left` and right subtree `right`. For example, the term

```
tree(1,tree(2,void,void),tree(3,void,void))
```

represents the tree with root 1 and children 2 and 3.

We call a binary tree `tree(x,left,right)` *nice* iff both of the following statements hold:

1. If `left` is not empty, then `x` is greater then all elements in `left`, and
2. If `right` is not empty, then `x` is less then all elements in `right`.

A binary tree is called a *search tree* iff each of its sub-trees is nice.

Write a program which tests whether a ground term is a search tree.

Hint: Use the following predicate in the definition of search tree:

```
is_search_tree(void).
is_search_tree(T) :- is_search_tree(T,Min,Max).
```

where `Min` and `Max` are the minimum and maximum element of the tree `T`. Then, implement the predicate `is_search_tree(T,Min,Max).`

Question 2 Write a program in Prolog to find a length of a given list. For example, goal `length([a, b, c, d, e])` should print `5`.

Question 3 Write a Prolog program which describes the directed graph in Figure 1. Define the path relation `path(Node1,Node2) :- ...` on this graph.

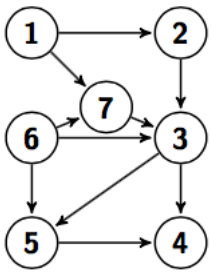

Figure 1: Directed graph.