German University in Cairo
Media Engineering and Technology
Prof. Dr. Slim Abdennadher

CSEN403: Concepts of Programming Languages, Spring 2015
Prolog Project Hints and Clarifications 1

Logistics Tips

- Team code: you can find your team code in the list of teams document posted on the MET website with the course material.
- Multiple submissions: You can resubmit the project by sending a new email with the new submission attached. You have to always use the correct subject and file names in all submissions. Please note that any submission after **the deadline (11:59 pm)** will **not** be considered. You must respect the deadline. If you think you will be working on the code and/or report till the last minute, we strongly advise you to make an earlier submission to be on the safe side. Then you can resubmit once you make better progress. **Avoid waiting till the last minute to make a first submission**. Being stressed, having Internet connectivity problems, etc are not acceptable excuses for missing the deadline.
- Time management: the project is not broken down to smaller milestones with multiple deadlines. It is part of the challenge that you learn with your team to make incremental progress every couple of days. You will need time to think, learn to express your approach in Prolog, search for built-in predicates, ask TAs, debug and test. You are highly discouraged from trying to dedicate non-top work days to finish the project just before the deadline. Instead, start early and spend few hours every couple of days working in a thoughtful way.

Technical Tips

- Branch Names: You may copy and use the following helper to complete your task in part b. The predicate `branch_name(S,N)` succeeds when `S` is a positive Integer representing the number of carbon atoms in a branch and `N` is an atom representing the corresponding branch name.

  ```
  branch_name(S,N):- HS is S*2+1,
              atomic_list_concat([c,S,h,HS],N).
  ```

  For instance:

  ```
  ?- branch_name(2,N).
  N = c2h5 ;
  false.
  ```

- How to generate a branched alkane? You can follow this systematic approach: Decide the length of the longest straight chain of that branched alkane and it will automatically determine for you how many carbon atoms should go into branches. For instance, if you need to generate a branched alkane of `X` carbon atoms, you can try to have:
  - A longest chain of `X-1` carbon atoms and `1` carbon atom in the branches.
  - Or a longest chain of `X-2` carbon atoms and `2` carbon atoms in the branches.
  - and so on ...

We recommend that you define some helpers that do basic tasks you need to perform. The following points present some ideas you may find useful.

- Helper ideas for part b,
  - The predicate `add_branch_to_carbon(InC, BSize, ResC)` which succeeds if an alkyl substituent branch of size `BSize` carbon atoms can be added to the structure of InC replacing a hydrogen atom. A carbon atom is allowed to have a down branch attached only if it already has a top branch attached. Test your predicate using the following sample queries:

    ```
    ?- add_branch_to_carbon(carb(c,h,h,c),2,Ans).
    Ans = carb(c, c2h5, h, c) ;
    false.

    ?- add_branch_to_carbon(carb(c,h,h,c),2,carb(c, h, c2h5, c)).
    false.

    ?- add_branch_to_carbon(carb(c,c2h5,h,c),2,Ans).
    Ans = carb(c, c2h5, c2h5, c) ;
    false.
    ```
  - The predicate `break_down(N, L)` which holds according to the following conditions: `N` is a positive integer. It represents the number of carbon atoms that will not be on the longest chain and need to be broken down into substituent groups. `L` is sorted list of positive integers that sum to N. Test your predicate with the following sample queries:

    ```
    ?- break_down(1,L).
    L = [1] ;
    false.
    ?- break_down(4,L).
    L = [1, 1, 1, 1] ;
    L = [1, 1, 2] ;
    L = [1, 3] ;
    L = [2, 2] ;
    L = [4] ;
    false.
    ```
- Be sure to check the built-in list predicates in Prolog. You may find some of them useful for your project. http://met.guc.edu.eg/Courses/Links.aspx?crsEdId=483