More Prolog Mini Exercises
Derivation Trees; Difference Lists; Controlling Search; CLPR

These questions use the Prolog rules in the lecture notes (both the basics and the ones on controlling search).

1. Draw the simplified derivation tree for the following goal:

   ```
   ?- reverse([1],R).
   ```

2. Consider `mymember` and also the `member_cut` rule defined in the notes on controlling search. What are all the answers that Prolog returns for the following goals?

   ```
   ?- mymember(1,[A,B,C]).

   ?- member_cut(1,[A,B,C]).
   ```

3. What are all the answers that Prolog returns for the following goals?

   ```
   ?- mymember(X,[1,2]), mymember(X,[0,2,2]).

   ?- member_cut(X,[1,2]), mymember(X,[0,2,2]).

   ?- mymember(X,[1,2]), member_cut(X,[0,2,2]).

   ?- member_cut(X,[1,2]), member_cut(X,[0,2,2]).
   ```

4. What are all the answers that Prolog returns for the following goals?

   ```
   ?- not(mymember(1,[1,2,3])).

   ?- not(mymember(5,[1,2,3])).

   ?- not(mymember(X,[1,2,3])).

   ?- mymember(X,[1,2,3]), not(mymember(X,[1,2,4])).

   ?- not(mymember(X,[1,2,4])), mymember(X,[1,2,3]).
   ```

5. Consider the standard version of `append`:

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

   If you know that the first argument is ground (that is, fully instantiated, containing no variables), there is a more efficient version that you can write by including a cut.

   (a) Define such a version.

   (b) Give an example of a query that has exactly the same behavior for both the standard version and the version with a cut.

   (c) Give an example of a query that behaves differently for for the standard version and the version with a cut.

   (d) What restrictions do we need on the inputs for the two versions to behave exactly the same? (Is it that the first argument is ground?)

6. Which of the following lists represent valid difference lists? For valid difference lists, what list do they represent?

```
[1,2|T]\T
[1,2,3]\[]
[1,2,3]\[1,2]
[1,2,3|T]\[3|T]
[1,2,3]\[1,2,3]
```

7. Write the list `[squid,clam]` as a difference list (in the most general possible way). Also draw a box-and-arrow diagram of the first part of the difference list.

8. Using the clpr library, write a rule `mymin` such that if you call `mymin(A,B,C)`, `C` will be the minimum of `A` and `B`.

9. Write a rule `solve` using the clpr library that solves the simultaneous equations $2x + 3y = 8$ and $x + y = 3$.

10. Again using the clpr library, write a rule `sum` such that for `sum(Xs,S)`, `S` is the sum of the numbers in the list `Xs`. You can assume the list consists only of numbers. For example `sum([],S)` should succeed with `S=0.0`, `sum([3,4],S)` should succeed with `S=7.0`, and `sum([A,A],10)` should succeed with `A=5.0`.