Week 13
Using Prolog

A language designed for A.I.

COMP 378 -- Spring, 2016
Mr. Weisert

Entering Prolog code from the console
1. Type at the Prolog prompt
   | ?- [user].
2. Prolog responds:
   compiling user for byte code...
3. Then enter your program: Here’s a very simple one:
   successor('Roosevelt','Truman').
   successor('Eisenhower','Kennedy').
   successor('Truman','Eisenhower').
   successor('Kennedy','Johnson').
   successor('Johnson','Nixon').
   predecessor(X,Y) :- successor(Y,X).
4. Escape with CTL-D
5. Prolog responds:
   user compiled, 6 lines read

Posting queries

■ A specific case:
   | ?- successor('Truman',Y).
     Y = 'Eisenhower'
     yes
■ A general test:
   | ?- successor(X,Y).
     X = 'Roosevelt'
     Y = 'Truman' ? ;
     X = 'Truman'
     Y = 'Eisenhower' ? ;
     X = 'Eisenhower'
     Y = 'Kennedy' ? ;
     X = 'Kennedy'
     Y = 'Johnson' ? ;
     X = 'Johnson'
     Y = 'Nixon'
(16 ms) yes

Note the semicolons

A couple of other tests

■ That was such a simple case, that there’s not much to test, but here’s another small demonstration showing matched bindings:
   | ?- successor('Roosevelt',X),successor(X,Y).

What would you expect Prolog to respond to these three queries?

■ ?- successor(X,'Truman').
■ ?- successor('Truman',X), predecessor(X,Y).
A couple of other tests

That was such a simple case, that there's not much to test, but here's another small demonstration showing matched bindings:

```
?- successor('Roosevelt',X),successor(X,Y).
   X = 'Truman'
   Y = 'Eisenhower'
?- successor(X,'Truman').
   X = 'Roosevelt' ? ; no
?- successor('Truman',X), predecessor(X,Y).
   X = 'Eisenhower'     Y = 'Truman' ? ; no
```

A severe limitation

Entering our whole program at the [user] prompt was fine for getting started with a trivial demonstration, but it's impractical for anything longer or anything intended to be permanent.

Here's a normal sequence for GNU Prolog 1.4.4.

1. Compose the program with an editor.
   - (You can and should now pay more attention to readability and commentary than was practical with the [user] interface.)
2. Store it in a .pl file
3. Compile it (If you picked the right options when you downloaded and installed GNU Prolog, you just need to double click on the above file.)
4. The REPL will appear with the program ready for inquiries.

Here's the first version

```
/* Presidential succession */
/* A simple demonstration program for COMP 378 */
/* Data base */
successor('Roosevelt','Truman').
successor('Eisenhower','Kennedy').
successor('Truman','Eisenhower').
successor('Kennedy','Johnson').
successor('Johnson','Nixon').
successor('Nixon','Ford').
successor('Ford','Carter').
successor('Carter','Reagan').
successor('Reagan','Bush41').
successor('Bush41','Clinton').
/* A Few Rules */
predicate(X,Y) :-successor(Y,X).
predecessor(X,Y) :-successor(Y,X).
earlierThan(X,Y) :-successor(X,Y). earlierThan(X,Y) :-successor(X,W),earlierThan(W,Y).
```

Some output from the stored version

```
?- earlierThan(X,'Truman').
X = 'Roosevelt' ? ;
no
?- earlierThan(X, 'Truman').
```
A recursive case

?- earlierThan('Kennedy', Z).
Z = 'Johnson' ;
Z = 'Nixon' ;
Z = 'Ford' ;
Z = 'Carter' ;
Z = 'Reagan' ;
Z = 'Bush41' ;
Z = 'Clinton' ;
no

Each time we hit enter after the semicolon Prolog gives us another solution.

Note:
- The examples we just saw were captured from actual Prolog interaction.
  - Therefore, you can assume that they're correct and will work as shown.
- Why did we show presentation slides instead of just demonstrating the examples live?
- Some of the later examples may not have been captured that way.
  - Should we trust them?

A little confusion!
- We weren’t entirely consistent in naming our predicates.
- We’ve been advised that a 2 parameter relationship should describe the relation of the first parameter to the second. But the English language is still ambiguous!
- So should successor(X, Y) mean:
  a. X has Y as a successor, or
  b. X is a successor of Y?
- We chose a, but many would argue for b.
- What about earlierThan?

Recommendation
- Try to pick unambiguous names.
  - successor might be ambiguous
  - earlierThan is less ambiguous
  - parent, child, father, mother, etc. generally describe the relationship of the first parameter to the second.
- Be as consistent as possible, especially on a project with multiple programmers.
An irritating shortcoming (especially at first)

- Some of GNU Prolog's error messages may be rather cryptic.
  - You know something is wrong, but you're not sure exactly what.
  - You may have to rely on a little trial & error diagnosis.

- As we discovered with Cloture, this is partly an issue of unfamiliarity.
  - Expect to get used to the interface as you gain experience with it.
  - Avoid making the same mistakes again!

Two minor Prolog errors made by your instructor that were hard to diagnose

1. Using double quote to delimit a string successor("Kennedy","Johnson").
   A habit picked up from other programming languages.

2. Following an old textbook that told us that semicolon (;) rather than percent (%) was the commentary delimiter.
   What would you guess would happen for each of those errors?

Should we have used character string constants at all?

- Most Prolog experts prefer to use named constants rather than quoted names.
  
  successor(truman,eisenhower).

- The choice depends on what we plan to do with the objects.