How’s it going?

Draw a picture that describes your first five weeks at HMC.
“Quizam"

- Take-home
- Available this Saturday evening at the latest
  Due Friday, October 10 at 5pm.
  Details coming on where to pick it up and drop it off
- Timed: 75 minutes
- One page of notes (two-sided)
- Two assignments & Grace Hopper overlap with the exam
You-try! member

```prolog
member( , ).
member( , ) :- ...
```

?- member(3, [1,2,3,4,5]).
true

?- member(6, [1,2,3,4,5]).
false.

?- member(E, [1,2,3,4,5]).
E = 1 ;
E = 2 ;
E = 3 ;
E = 4 ;
E = 5.
You-try! `member`

`member` is built in to SWI-Prolog

```prolog
member(E, [E|_]).
member(E, [_|R]) :- member(E, R).
```

?- member(3, [1,2,3,4,5]).
true;
false.

?- member(6, [1,2,3,4,5]).
false.

?- member(E, [1,2,3,4,5]).
E = 1 ;
E = 2 ;
E = 3 ;
E = 4 ;
E = 5.
The “Zebra” Puzzle

a.k.a. The “Einstein” Puzzle

five nationalities
norwegian, brit, swede, dane, german

five pets
dog, bird, zebra, cat, horse

five cigars
pallmall, winfield, dunhill, rothmans, marlboro

five beverages
tea, coffee, milk, water, beer

five house colors
red, green, yellow, blue, white

fifteen clues
(1) The norwegian lives in the first house.

Who owns the zebra?
The “Zebra” Puzzle

all the clues

(1) The Norwegian lives in the first house.
(2) The person living in the center house drinks milk.
(3) The Brit lives in a red house.
(4) The Swede keeps dogs as pets.
(5) The Dane drinks tea.
(6) The Green house is next to, and on the left of the White house.
(7) The owner of the Green house drinks coffee.
(8) The person who smokes Pall Mall rears birds.
(9) The owner of the Yellow house smokes Dunhill.
(10) The man who smokes Marlboro lives next to the one who keeps cats.
(11) The man who keeps horses lives next to the man who smokes Dunhill.
(12) The man who smokes Winfields drinks beer.
(13) The German smokes Rothmans.
(14) The red house is to the right of the blue.
(15) The Norwegian doesn't live by the red, white, or green houses.
The “Zebra” Puzzle

Representation in Prolog: all possible houses, i.e., the “state space”

houses( [ H1, H2, H3, H4, H5 ]):-

    H1 = [ N1, P1, S1, B1, C1 ],
    H2 = [ N2, P2, S2, B2, C2 ],
    H3 = [ N3, P3, S3, B3, C3 ],
    H4 = [ N4, P4, S4, B4, C4 ],
    H5 = [ N5, P5, S5, B5, C5 ],

    perm( [N1,N2,N3,N4,N5], [norwegian, brit, swede, dane, german] ),
    perm( [P1,P2,P3,P4,P5], [dog, bird, zebra, cat, horse] ),
    perm( [S1,S2,S3,S4,S5], [pallmall, winfield, dunhill, rothmans, marlboro] ),
    perm( [B1,B2,B3,B4,B5], [tea, coffee, milk, water, beer] ),
    perm( [C1,C2,C3,C4,C5], [red, green, yellow, blue, white] ).

H1  H2  H3  H4  H5
The "Zebra" Puzzle

Representation in Prolog: the clues

einstein(Houses) :-
  Houses = [[norwegian, _, _, _, _], _, [_, _, _, milk, _], _, _],
  member([brit, _, _, _, red], Houses),
  ... 
houses( Houses ), % it's important to have this LATE (otherwise it's 120**5)
  ...

  (1) The Norwegian lives in the first house.
  (2) The person living in the center house drinks milk.
  (3) The Brit lives in a red house.
  ...
The “Zebra” Puzzle

Representation in Prolog: solving the puzzle

solve :-
    einstein( [ H1, H2, H3, H4, H5 ] ),
    write( ' first house: '), write(H1), nl, % nl is "newline"
    write( ' second house: '), write(H2), nl,
    write( ' third house: '), write(H3), nl,
    write( ' fourth house: '), write(H4), nl,
    write( ' fifth house: '), write(H5), nl.
What’s This?

\[
d(X, X, 1) :- !.
\]

\[
d(C, _, 0) :- number(C).
\]

\[
d(X^C, X, C*X^(C-1)) :- number(C).
\]

\[
d(F+G, X, U+V) :- d(F,X,U), d(G,X,V).
\]

\[
d(F-G, X, U-V) :- d(F,X,U), d(G,X,V).
\]

\[
d(F*G, X, (U*G)+(V*F)) :- d(F,X,U), d(G,X,V).
\]

\[
d(F/G, X, ((U*G)-(V*F))/(G^2)) :- d(F,X,U), d(G,X,V).
\]

?- d(x, x, R).
R = 1.

?- d(3*x, x, R).
R = 0*x+1*3.
You could write a simplifier!

?- d(x, x, R).
R = 1.

?- d(3*x, x, R).
R = 0*x+1*3.

www.cs.hmc.edu/cs42/code/symdiff.pl

diff(Term, Var, Result) :- d(Term, Var, Term2), simplify(Term2, Result).

:- begin_tests(diff).
test(multrule_1) :- diff(3*x, x, 3).
test(multrule_2) :- diff(3*4, x, 0).
test(div_rule) :- diff(3*x/x, x, 0).
test(exp_rule) :- diff(x^2, x, 2*x).
:- end_tests(diff).
1.1 Syntax of the Predicate Calculus

**Definition.** The syntax of the predicate calculus ($\mathcal{PC}$) consists of symbols and formulas as follows:

**Symbols**

- Parentheses: $(, )$
- Sentential connectives: $\neg, \lor, \land, \rightarrow, \leftrightarrow$
- Quantifiers: $\forall, \exists$
- SC letters (sentential letters): $A, B, \cdots, Z$, and any of these letters with a positive Arabic numeral subscript.

**Predicate symbols:** An $n$-ary predicate is an uppercase letter, $A, \cdots, Z$, with the numeral $n$ as a superscript, where $n$ denotes the arity of the predicate and $0 < n$. These uppercase letters may also have numerical subscripts. Note: We will usually omit the superscript when we know the arity of a predicate.

**Individual constants:** Lowercase letters $a, \cdots, r$, with or without numerical subscripts.

**Individual variables:** Lowercase letters $s, \cdots, z$, with or without numerical subscripts.

**Formulas**

The set of all predicate calculus ($\mathcal{PC}$) formulas is defined recursively, beginning with the atomic formulas.

**Atomic Formula:**
Any single SC letter, or an $n$-ary predicate followed by exactly $n$ symbols, each of which is either an individual constant or a variable.

**Formula:**
Any atomic formula, or any expression (finitely long string of symbols) that is obtainable by use of the following predicate calculus construction rules (PCCR):
Prolog is syntactic sugar for the predicate calculus.
Prolog

(proofs in)
Predicate Calculus
p :- p.

Racket

λ

(\(\lambda (x) (x x)) \ (\lambda (x) (x x)) \)

Church (1936), Turing (1937)