Prolog: Implementation and Design Issues

Mark Neal

Skinning Cats

- As usual, there are many ways...
- Not all ways are equal in Prolog
  (or any other programming language!)
- Some are easier to read
- Some are more efficient
  - Factorial examples

“Design”

- Think hard about what you want to do:
  - What information do you need to store?
  - How are you going to store it?
  - How can you encode useful relationships within it?
  - What exactly do you want Prolog to figure out for you?
  - How can you use Prolog's inference mechanisms to solve the problem for you?
  - Can you do things without lots of unnecessary calculation?
- Try to avoid shoe-horning a Java/C++ (or whatever) design into Prolog

Implementation

- Obvious tools:
  - Lists
  - Lists of lists
  - Lists of lists of lists...
  - Recursion
  - Well thought through queries