Introduction to Prolog

A logic-based programming language

Logic programming paradigm

- Programmer specifies relationships among data values
- User poses queries

Comparison with the functional paradigm (Clojure, etc.)

- A functional program consists of a sequence of function definitions.
- A logic program consists of a sequence of relation definitions.
- "The Prolog execution environment doesn't so much compute an answer, it deduces" an answer from the relation definitions at hand. Rather than being given an expression to evaluate, the Prolog environment is given an expression which it interprets as a question."
  -- Bucknell University Prolog manual

Prolog is different from other programming languages

- Prolog has no types. In fact, the basic logic programming environment has no literal values as such.
- Identifiers starting with lower-case letters denote data values (almost like values in an enumerated type) while all other identifiers denote variables.
- Though the basic elements of Prolog are typeless, most implementations have been enhanced to include character and integer values and operations.
- Also, Prolog has mechanisms built-in for describing tuples and lists.
  -- ibid

See
Backward chaining

- The query asks:

"For what parameter values [variables in the expression] does the expression evaluate to true?"

Prolog versus Lisp
(i.e. backward versus forward chaining search emphasis)

- For reasons that aren't altogether clear
  - A.I. researchers in America prefer Lisp and Lisp-like languages
  - A.I. researchers in Europe and Asia prefer Prolog and Prolog-like languages.

- But they will all use either approach when the problem clearly calls for it.