Logic Programming

- Four main programming paradigms:
  - imperative,
  - object-oriented,
  - functional,
  - logical programming.
- Symbolic logic provides basis for logic programming
- **Prolog** (from *Pro*gramming in *Log*ic) is a programming language developed in the 1970s by researchers in artificial intelligence (AI).

The Origins of Prolog

- 1970s:
  - University of Aix-Marseille (Alain Calmerauer & Phillippe Roussel)
    - Developed fundamental design of Prolog
    - Used it for NLP (natural language processing)
  - University of Edinburgh (Robert Kowalski)
    - Helped to design Prolog
    - Used it for automated theorem proving
- 1981-1993:
  - Fifth Generation Computing Systems project launched by Japanese government
    - Objective: to develop intelligent machines (AI) using Prolog

Main Applications of Logic Programming

- Relational database management systems
- Expert systems
- Natural language processing

Basic Facts about Logic Programming

- ***Logic Paradigm*** **takes a declarative approach to problem-solving. Various logical assertions about a situation are made, establishing all known facts. Then queries are made. The role of the computer becomes maintaining data and logical deduction.**

- ***Logic Paradigm Programming:***
  A logic program is divided into three sections:

  - **a series of definitions/declarations that define the problem domain**
  - **statements of relevant facts and rules**
  - **statement of goals in the form of a query**

Logic Programming

- Prolog programs include *Prolog facts* and *Prolog rules*.
- As an example of a set of Prolog facts consider the following:

```
instructor(chan, math273).
instructor(patel, ee222).
instructor(grossman, cs301).
enrolled(kevin, math273).
enrolled(juana, ee222).
enrolled(juana, cs301).
enrolled(kiko, math273).
enrolled(kiko, cs301).
```

- Here the predicates *instructor(p,c)* and *enrolled(s,c)* represent that professor *p* is the instructor of course *c* and that student *s* is enrolled in course *c*.

Logic Programming (cont)

- In Prolog, names beginning with an uppercase letter are variables.
- If we have a predicate *teaches(p,s)* representing "professor *p* teaches student *s*," we can write the rule:

```
teaches(P,S) :- instructor(P,C), enrolled(S,C).
```

- Prolog rule can be viewed as equivalent to the following statement in logic (using our conventions for logical statements).

$$\forall p\ \forall c\ \forall s(I(p,c) \wedge E(s,c)) \rightarrow T(p,s))$$

Logic Programming (cont)

- Prolog programs are loaded into a *Prolog interpreter*. The interpreter receives *queries* and returns answers using the Prolog program.
- For example, using our program, the following query may be given:

```
?enrolled(kevin,math273).
```

- Prolog produces the response:

```
yes
```

- Note that the `?` is the prompt given by the Prolog interpreter indicating that it is ready to receive a query.

Logic Programming (cont)

- The query:

```
?enrolled(X,math273).
```

produces the response:

```
X = kevin;
X = kiko;
no
```

- The query:

```
?teaches(X,juana).
```

produces the response:

```
X = patel;
X = grossman;
no
```

The Prolog interpreter tries to find an instantiation for X. It does so and returns `X = kevin.` Then the user types the `;` indicating a request for another answer. When Prolog is unable to find another answer it returns `no`.

Logic Programming (cont)

- The query:

```
?teaches(chan,X).
```

produces the response:

```
X = kevin;
X = kiko;
no
```

Logic Programming (cont)

A number of good online introductions to Prolog are available:

- https://bernardopires.com/2013/10/try-logic-programming-a-gentle-introduction-to-prolog/
- http://www.learnprolognow.org/