Introduction to Turbo Prolog

Professor Dr. A. K. M. Akhtar Hossain

Dept. of CSE, University of Rajshahi

History of Prolog

- The name **Prolog** was taken from the "**Pro**gramming in **Log**ic".

- In 1972, Alain Colmerauer and P. Roussel were originally developed Turbo Prolog programming language at the University of Marseilles in France.

Applications for Turbo Prolog

- Expert Systems
- Natural language Processing
- Robotics
- Gaming
- Simulations

Starting Turbo Prolog

- Turbo Prolog have for window:
  - ❖Editor Window (Create or edit Program)
  - ❖Massage Window (Keeps processing activity)
  - ❖Dialog Window (Show Output)
  - ❖Trace Window (Finding Problems)

Menu bar of Turbo Prolog

- Run
- Compile
- Edit
- Options
- Files
- Setup
- Quit

Turbo Prolog Domain/Data Types:

char	Single character (enclosed between single quotation marks).
integer	Integer from -32,768 to 32,767
real	Floating-point number ($1e^{-307}$ to $1e^{308}$)
string	Character sequence (enclosed between double quotation marks)
symbol	Character sequence of letters, numbers and underscores, with the character a lowercase letter.
file	Symbolic file name

Create a Simple Program

- Select the Edit mode and enter the program.
- Enter text or make corrections as necessary.
- Save the program to disk using Files save option.
- Compile the program.
- Execute/Run the program.

Keys Used In the Editor Window

- Ctrl+KB: mark start of block
- Ctrl+KK: mark end of block
- Ctrl+KC: copy marked block
- Ctrl+KY: delete marked block
- Ctrl+KV: move marked block
- F1: help on key usage

PROLOG.EXE

Files Edit Run Compile Options Setup

Editor

Line 1 Col 1 WORK.PRO Indent Insert

```
/* Simple Test Program */
predicates
            likes(symbol, symbol)
clauses
        likes(frank,sue).
        likes(harold,ruth).
        likes(donad,lee).
```

Dialog

```
Goal: likes(harold,lee)
No
Goal: _
```

Message

```
Load WORK.PRO
Compiling WORK.PRO
Compilation successful
likes
```

Trace

F2-Save F3-Load F5-Zoom F6-Next F8-Previous goal Shift-F10-Resize F10-End

PROLOG.EXE

Files Edit Run Compile Options Setup

Editor

Line 2 Col 11 F:/PROLOG/TEST2.PRO Indent Insert

```
/* Example Only */
/*  Not for Medical Use */

  domains
          disease,indication = symbol

  predicates
           symptom(disease,indication)

  clauses
         symptom(chicken_pox,high_fever).
         symptom(chicken_pox,chills).
         symptom(flu,chills).
         symptom(cold,mid_body_ache).
```

Message

```
Press the SPACE bar
Compiling F:/PROLOG/TEST2.PRO
symptom
```

Trace

F1-Help F2-Save F3-Load F5-Zoom F6-Next F7-Xcopy F8-Xedit F9-Compile F10-Menu

How to Install Turbo Prolog in MS Windows 7/10

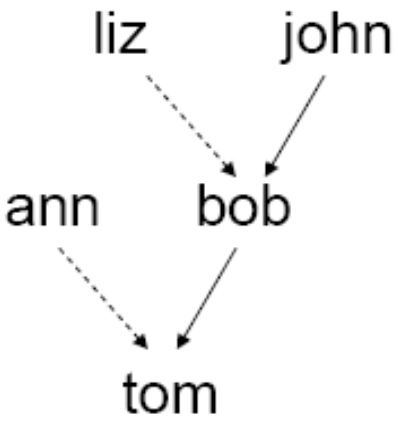
- Turbo Prolog installation Process: YouTube link.
- https://www.youtube.com/watch?v=gEa1dL9Iwv0
- Use Notepad to write Turbo Prolog Codes.
- Save the program.
- **Examples:**
    - **Test1.pro**
    - **Test2.pro**

Expressing Facts

- **Logic programming deals with relations rather than functions**

facts

```
father(tom, bob).
father(bob, john).
mother(tom, ann).
mother(bob, liz).
```



Relations and Facts

- **Relations**
  - represented by n-tuple $(a_1, a_2, \ldots, a_n)$
- fact: **all a_i's are <u>concrete terms</u>**
  - <u>concrete term</u> begins with a lower-case letter (symbols)

symbols, numbers, …

father(tom, bob).
father(bob, john).

a predicate

predicate name indicates which relation

the tuple indicates what objects are involved in the relation

tom, bob, john are all symbols.

Turbo Prolog Rules

- **<u>Rules:</u>**
- A rule is an expression that indicates that the truth of a particular fact depends upon one or more other facts.
- **Example:**

If there is a body stiffness or pain in the joints

AND there is a sensitivity to infections,

Then there is probably a vitamin C deficiency.

Turbo Prolog Rules Conti…

- This rule could be expressed as the following Turbo Prolog clause:
- hypothesis(vitc_deficiency) if

  symptom(arthritis) and

  symptom(infection_sensitivity).

In this abbreviated form the previous rule becomes:

- hypothesis(vitc_deficiency):-

  symptom(arthritis),

  symptom(infection_sensitivity).

Rules

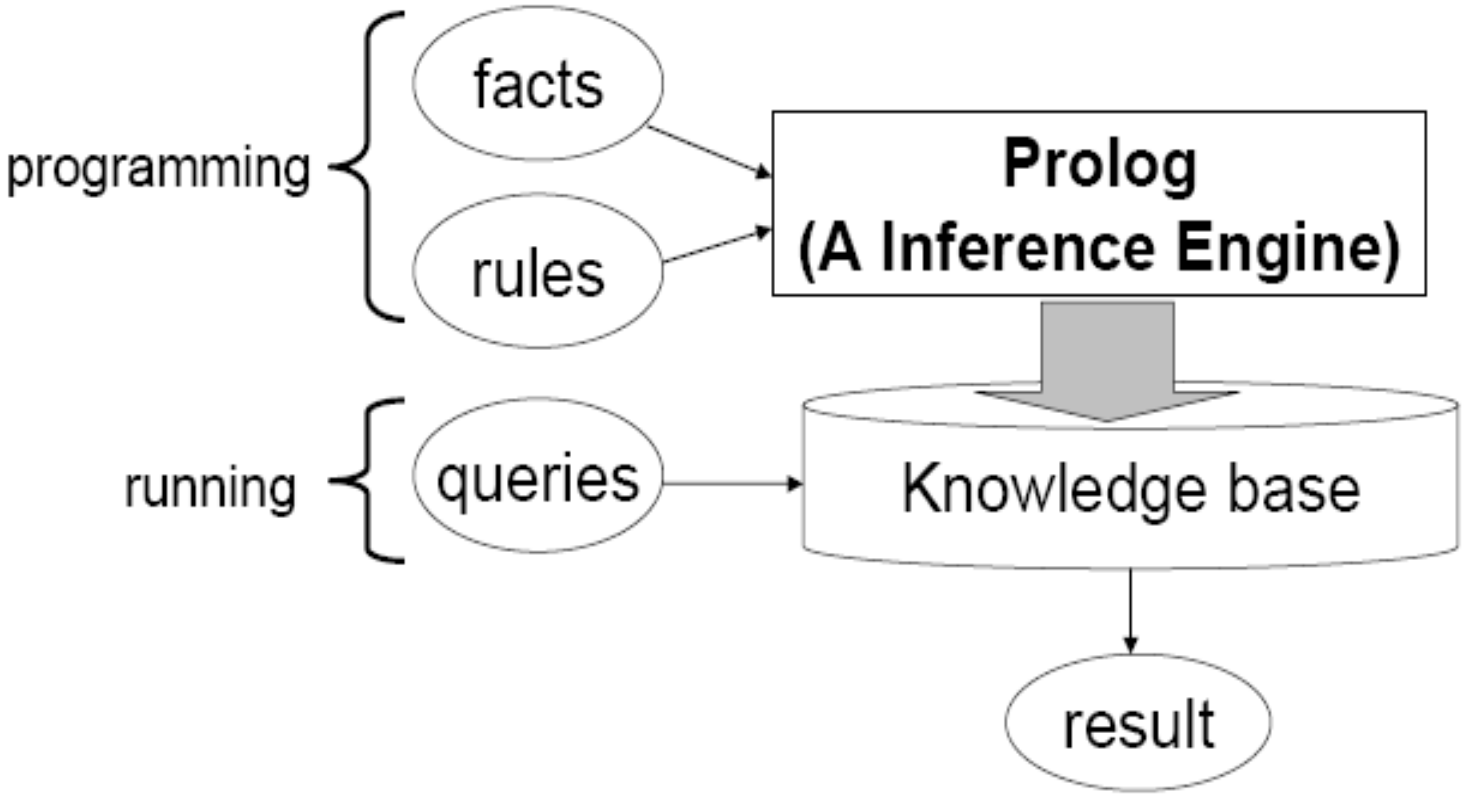
- **some relation or part of a relation can be defined with** rules
  - ☞ P if Q_1 and Q_2 and … and Q_k.

$$P \text{ :- } Q_1,\ Q_2,\ \ldots,\ Q_k.$$

  - ☞ capitalized symbol represents variables

```
grandfather(X, Y) :- father(X, Z), father(Z, Y).
likes(tom, Which) :- likes(john, Which).
```

Prolog Environment



Turbo Prolog Data Types

- <u>Basically two Types:</u>
- **Variables**
- **Non-variables:**

  I. **Objects**

  II. **Others**

I. Objects are (six types) : symbol, string, integer, real, char, file.

II. Others: List, Compound Object.

- Domains
  - type definition (optional)

symbol, integer, and real are built-in types.

```
domains
   brand, color = symbol
   age, price = integer
   mileage = real
```

+

- Predicates
  - predicate declaration

```
predicates
   car(brand,mileage,age,color,price)
```

predicate: name of relation

||

```
predicates
   car(symbol,real,integer,symbol,integer)
```

cannot be run under 64-bit Win7

Program structure

```
domains
   ...
predicates
   ...
goal
   ...
clauses
   ...
```

- Goal
  - provide queries (optional)
- Clauses
  - describe facts and rules in predicate form
  - dot (“.”) is used to separate clauses

Some Syntax Rule

- names of objects and predicates begin with a lower-case letter
- variable names are capitalized
- anonymous variables ("_") can be used wherever a variable can be used

```
likes(tom,_).    /* tom likes everyone */
lives(_).        /* everyone lives */
```

- comments are enclosed within "/*" and "*/"

More Menus

- Compile Menu
  - compile to OBJ/EXE file (F9)
- switch between windows (F6)
- zoom in/out the current window (F5)
- exit Turbo Prolog (Alt-X)

```
/* Example Only */
 /* Not for Medical Use */
 domains
      disease, indication = symbol
 predicates
       symptom(disease, indication)
 clauses
      symptom(chicken_pox,high_fever).
      symptom(chicken_pox,chills).
      symptom(flu,chills).
      symptom(cold,mid_body_ache).
      symptom(flu,severe_body_ache).
      symptom(cold,runny_nose).
      symptom(flu,runny_nose).
      symptom(flu,moderate_cough).
```

GOAL WINDOW:

- Goal:symptom(cold,runny_nose) ┘
- True
- Goal:
-

Goals

- Goal: Queries about relations
  - whether a particular tuple belongs to a relation?

    contains no variables
    - yes/fail answers rather than yes/no
    - "fail" means "not found"
  - queries containing variables
    - a request for suitable values for the variables in the relation

Prolog Vs. Turbo Prolog

Words	Turbo Prolog	Prolog / Turbo Prolog
If	:-	if
And	,	and
Or	;	or
Query	?-	Goal:

How to Trace a Program?

① Add a compiling directive trace in your program

```
trace
predicates
   likes(symbol, symbol)
clauses
   …
```

② run the program. Input the goal

```
Goal: likes(bill, X)
```

③ Press F10 to step

☞ trace result is shown in the trace window

Practice

- trace the goal `likes(bill, X)` with the following program

```
clauses
  likes(ellen, tennis).
  likes(john, football).
  likes(tom, baseball).
  likes(eric, swimming).
  likes(mark, tennis).
  likes(bill, Activity) if likes(tom, Activity).
```

Turbo Prolog Free and Bound Variables:

Free and Bound Variables

- Free variable
  - a variable without known value
- bound variable
  - a variable whose value is known

Turbo Prolog Variable

- A variable name must begin with capital letter.
- Example:
  - Age
  - Disease
  - Patient
  - John

Book:

- Introduction to *Turbo Prolog*.
- Author: *Carl Townsend*.
- Publication. New Delhi: BPB Publications. Publication.

```
/* Example Only */
 /* Not for Medical Use */
 domains
      disease, indication = symbol
 predicates
       symptom(disease, indication)
 clauses
      symptom(chicken_pox,high_fever).
      symptom(chicken_pox,chills).
      symptom(flu,chills).
      symptom(cold,mid_body_ache).
      symptom(flu,severe_body_ache).
      symptom(cold,runny_nose).
      symptom(flu,runny_nose).
      symptom(flu,moderate_cough).
```

Examples:

- Goal:Symptom(Disease,runny_nose) ┘

- Goal:
  - Disease = cold
  - Disease = flu
  - 2 solutions
- Goal:

Examples:

- Goal: symptom(cold, Symptom)
  - Symptom = mild_body_ache
  - Symptom = runny _nose
  - 2 Solutions
- Goal:

Examples for Anonymous variables

- Sometimes prolog ignore the value of one or more arguments when determining a goal's failure or success.
- Example:
- Goal: symptom(_,chills)
- **True**
- Goal:

Examples for Compound Goals

- Goal: symptom(Disease, mild_body_ache)**and** symptom(Disease, runny_nose)
- **Disease = cold**
- **1 Solution**
- Goal:

Turbo Prolog Unification

- Unification is a **pattern matching** process.
- A term is said to unify with another term if :-
- Both terms appear in predicates that have the **same number of arguments**, and both terms appear in the same position in their predicates.
- Both terms appear as **arguments of the same type – a symbol type** can only unify with a symbol type and so on.
- All subterms unify with each other.

Turbo Prolog Unification Conti….

- A variable that is free will unify with any term that satisfies the preceding conditions. After unification, the variable is bound to the value of the term.
- A constant can unify with itself or any free variable. If the constant is unified with a variable, the variable will be bound to the value of the constant.

Turbo Prolog Unification Conti…

- A free variable will unify with any other free variable. After unifying the two variables will act as one. If one of the variables becomes bound, the other will be bound to the same value.

Turbo Prolog Unification Conti..

- Predicates unify with each other if :-
  - They have the same relation name.
  - They have the same number of arguments.
  - All argument pairs unify with each other.

Unification: Term Matching

- free variable can match any term.
  - constant, variable, list, compound terms…
- After unification, the variable is bound to that term

Match writen_by(X, Y) with
writen_by(fleming, book("Dr No", 210))

X = fleming, Y = book("Dr No", 210)

compound term

Unification Rule 2

- a constant can match itself or a free variable

book(“Moby Dick”, Y)

can match

book(“Moby Dick”, 210)

but cannot match

book(“Dr. No”, 201)

Unification Rule 3

- a compound term can match another compound term if and only if
  - they have the same number of arguments
  - we can match arguments pairwisely

writen_by(X, book("Moby Dick", Y))

cannot match

writen_by(fleming, book("Dr No", 210))

Turbo Prolog Input & Output Predicates:

- *Input* Predicates:
  - *readln*
  - *readint*
  - *readchar*
  - *readreal*
- Output Predicates:
  - write
  - writef
  - writedevice

Examples:

ReadLine and Write Predicate

```
address(City,"Ontario"):-
        write("Does the "+City+" belond to Ontario(yes/no)?"),
        readln(Reply),nl,
        write("Right answer is :").


address("Las Vegas","California").
address("London","Ontario").
address("Istanbul","Turkey").
```

Examples:

ReadInt and Write Predicate

```
chkage(Patient):-
        write("What is",Patient,"s Age?"),
        readint(Age),nl,
        Age>12,nl,
        Write("Patient cannot be evaluated."),nl.

chkage("James").
chkage("Alex").
```

Examples:

ReadReal and Write predicate

- askprice(Item,Price):-
- write("What is the price of ",Item,"?"),
- readreal (Price),nl,
- write(Item "Price is",Price).

Examples:

Writef predicate

- Formatted write. Format is an atom whose characters will be printed. Format may contain certain special character sequences which specify certain formatting and substitution actions. Arguments provides all the terms required to be output.

- likes(ahmet,cricket).
- likes(hicran,cricket).
- likes(ahmet,football).
- likes(hicran,snooker).
- likes(ahmet,computergames).
- likes(hicran,tennis).

- run:-
- writef("Enter Person Name="),
- read(Person),nl,
- likes(Person,X),
- writef("%t likes %t \n" , [Person,X]),
- fail.

- **End Today**