Simple Rule-Based Chatbot Agent Implementation in Prolog

1. To implement a rule-based system in Visual Prolog that makes driving decisions by evaluating traffic light status, obstacles, and speed based on the PEAS model.

```
DOMAINS
   light = symbol
   obstacle = symbol
   speed = integer
PREDICATES
   traffic_light(light)
   is_obstacle(obstacle)
   current_speed(speed)
  nondeterm drive_decision
CLAUSES
   % Environment settings
   traffic_light(green).
   is_obstacle(none).
   current_speed(40).
   % Decision-making based on PEAS
   drive_decision :-
      traffic_light(green),
      is_obstacle(none),
      current_speed(S), S < 60,
      write("Drive Forward - Road is Clear and Light is Green."), nl.
   drive_decision :-
      traffic_light(red),
      write("Stop - Red Light."), nl.
   drive_decision :-
      is_obstacle(car),
      write("Slow Down - Obstacle Ahead."), nl.
   drive_decision :-
      current_speed(S), S >= 60,
      write("Reduce Speed - Too Fast."), nl.
```

Goal

```
   %drive_decision.
   %traffic_light(L).
   %is_obstacle(O).
   %current_speed(S).
   traffic_light(L),
    write("Traffic light is: "), write(L), nl,
    is_obstacle(O),
    write("Obstacle: "), write(O), nl,
    current_speed(S),
    write("Speed: "), write(S), nl.
```

2. To implement a basic chatbot in Visual Prolog that provides fixed responses to specific user inputs using logical clauses.

DOMAINS

```
              input = string
```

PREDICATES

```
             Nondeterm respond(input)
```

CLAUSES

```
             respond("hello") :-
              write("Hi! How can I assist you today?"), nl.
             respond("how are you") :-
              write("I'm just a bunch of code, but I'm running smoothly!"), nl.
              respond("bye") :-
              write("Goodbye! Have a great day."), nl.
             respond(_) :-
              write("Sorry, I didn't understand that."), nl.
```

GOAL

```
 %respond("hello").
%respond("bye").
 respond("how are you").
%respond("What is your name").
```

3. To simulate a basic chatbot that interacts with user input.

Domains

```
chatLoop=symbol.
Input=string.
```

Predicates

```
nondeterm respond(Input)
nondeterm run
nondeterm chatLoop
```

Clauses

```
run() :-
  write("Welcome! Type your message (type 'bye' to exit)."), nl,
    chatLoop.
chatLoop() :-
    write("> "),
    readln(Input),
    Input = "bye",
    respond(Input),
    write("Chat ended."), nl.
chatLoop() :-
    write("> "),
    readln(Input),
    respond(Input),
    chatLoop().
respond("hello") :-
   write("Hi! How can I assist you today?"), nl.
respond("how are you") :-
    write("I'm just a bunch of code, but I'm running smoothly!"), nl.
respond("bye") :-
   write("Goodbye! Have a great day."), nl.
respond(_) :-
    write("Sorry, I didn't understand that."), nl.
```

Goal

```
 run.
%respond("hello").
%respond("bye").
%respond("Your Name").
  %respond("hello"), respond("how are you"), respond("bye").
```