PROJECT REPORT

Rock–Paper–Scissors Game – Using Prolog

Submitted By

Name: Choudante Apurwa Sudhir
Roll No.: 178
Branch: Computer Science & Engineering
Subject: Artificial Intelligence
Date: *27/11/2025*

Title of the Project

Rock–Paper–Scissors Game in Prolog

Abstract

This project implements the classic Rock–Paper–Scissors game using SWI-Prolog. The player selects an option—rock, paper, or scissors—while the computer generates a random move using Prolog's random library. The game then compares both choices based on predefined rules and displays whether the player wins, loses, or ties.
This project demonstrates interactive programming, rule-based decision-making, predicate logic, and user input/output handling in Prolog.

Objectives

- To develop an interactive AI-based game using Prolog.
- To understand Prolog predicates, rules, and pattern matching.
- To implement decision-making using logical rules.
- To perform user input/output operations using Prolog.
- To demonstrate simple AI reasoning through rule-based logic.

Software and Tools Used

Category	Description
Programming Language	Prolog
Interpreter / IDE	SWI-Prolog ≥ 8.0
Operating System	Windows / Linux
File Name	`rps_game.pl`
Method Used	Rule-based Logic + Randomization

Working Principle

1. The game displays the rules and prompts the user to begin.
2. User enters one among: **rock**, **paper**, or **scissors**.
3. Prolog generates a random move for the computer.
4. The game compares both moves using standard rules:
    - Rock beats Scissors
    - Scissors beat Paper
    - Paper beats Rock
5. The game displays Win, Lose, or Draw.
6. User is asked if they want to play again.
7. Program continues until user chooses to stop.

Flow of Execution

```
Start
   ↓
Display Instructions
   ↓
User enters move (rock / paper / scissors)
   ↓
Generate computer move (random)
   ↓
Compare both moves
   ├─ If user wins → Display "You Win!"
   ├─ If draw → Display "Draw!"
   └─ If computer wins → Display "You Lose!"
   ↓
Ask to play again
   ↓
If yes → Repeat
If no → End Program
   ↓
End
```

Code Implementation

```
:- use_module(library(random)).

choice(rock).
choice(paper).
choice(scissors).

start :-
    write('□ Rock-Paper-Scissors Game!'), nl,
    write('Choose: rock, paper, or scissors.'), nl,
    write('Type play. to begin!'), nl.

play :-
    random_choice(Computer),
    write('Your move: '),
    read(Player),
    ( valid(Player) ->
        format('Computer chose: ~w~n', [Computer]),
        result(Player, Computer)
    ;
        write('Invalid choice! Try again.'), nl,
        play
    ),
    ask_restart.

random_choice(C) :-
    findall(X, choice(X), L),
    random_member(C, L).

valid(X) :- choice(X).

result(Player, Player) :-
    write('Draw!'), nl.

result(rock, scissors) :-
    write('You win! Rock beats scissors.'), nl.

result(paper, rock) :-
    write('You win! Paper beats rock.'), nl.

result(scissors, paper) :-
    write('You win! Scissors cut paper.'), nl.

result(_, _) :-
    write('You lose!'), nl.

ask_restart :-
    write('Play again? (yes/no): '),
    read(Ans),
    ( Ans == yes ->
        nl, play
    ; Ans == no ->
        write('Thanks for playing!'), nl
    ;
        write('Type yes or no.'), nl,
```

```
        ask_restart
    ).
```

Execution Commands

Open SWI-Prolog and type:

```
?- ['C:/Users/.../rps_game.pl'].
?- start.
?- play.
```

Sample Output

```
Rock-Paper-Scissors Game!
Choose: rock, paper, or scissors.
Type play. to begin!

Your move: rock.
Computer chose: scissors
You win! Rock beats scissors.

Play again? (yes/no): yes.
```

Conclusion

The Rock–Paper–Scissors game demonstrates the fundamentals of logical programming using Prolog. Through rule-based reasoning, user interaction, and random choice generation, the project showcases how Prolog can be used to create AI-based interactive applications.
It provides a strong foundation for building more complex Prolog games and logic-driven systems.

Future Enhancements

- Adding a score counter.
- Creating difficulty levels.
- Adding a GUI version using Python or JavaScript.
- Making a multiplayer mode.
- Adding sound effects or animations.