Simple Game AI for Rock-Paper-Scissors

Problem Statement:

Develop an AI-powered **Rock-Paper-Scissors** game where the AI competes against a human using **random selection or a rule-based strategy**. The implementation will be done in **Python on Google Colab**, with a report detailing the approach, code, and results.

Submitted by:

Name: SHARAD KRISHNA SINGH
Roll Number: 202401100400170

Course: BTECH
Department: CSE AIML

Introduction

Rock-Paper-Scissors is a popular hand game played between two participants, where each player chooses one of three options: **Rock, Paper, or Scissors**. The game operates on simple rules:

- **Rock beats Scissors** (by crushing it).
- **Scissors beats Paper** (by cutting it).
- **Paper beats Rock** (by wrapping it).

Despite its simplicity, this game is often used to study **decision-making, probability, and AI strategies**. In this project, we aim to develop an **AI opponent** capable of playing Rock-Paper-Scissors against a human. The AI will make its choices based on either **random selection** or a **rule-based approach** that can improve its strategy over multiple rounds.

The challenge lies in making the AI more competitive rather than just making random choices. By implementing different strategies, such as **pattern recognition** or **weighted probabilities**, we can improve the AI's performance. This project provides insight into **basic artificial intelligence, game theory, and programming logic**.

Methodology: Approach to Solve the Problem

The Rock-Paper-Scissors AI game is designed using a structured approach to ensure smooth gameplay and AI decision-making. The key steps involved in solving the problem are:

1. **User Input Handling**
    - The user selects one of the three valid choices: **rock, paper, or scissors**.
    - Input validation ensures that the user enters a correct choice.
2. **AI Decision-Making**
    - In the first round, the AI **randomly selects** a move.
    - From the second round onwards, the AI follows a **counter-strategy** by predicting that the user might repeat their previous move and choosing the best counter-move.
3. **Winner Determination**
    - The game follows the standard Rock-Paper-Scissors rules to compare the user's choice with the AI's choice.
    - If both choices are the same, the game is a **tie**. Otherwise, the winner is decided based on the predefined rules.
4. **Game Loop & Replay Option**
    - After displaying the result, the user is given the option to **play again or exit** the game.
    - If they choose to continue, the AI remembers the previous choice for better predictions.

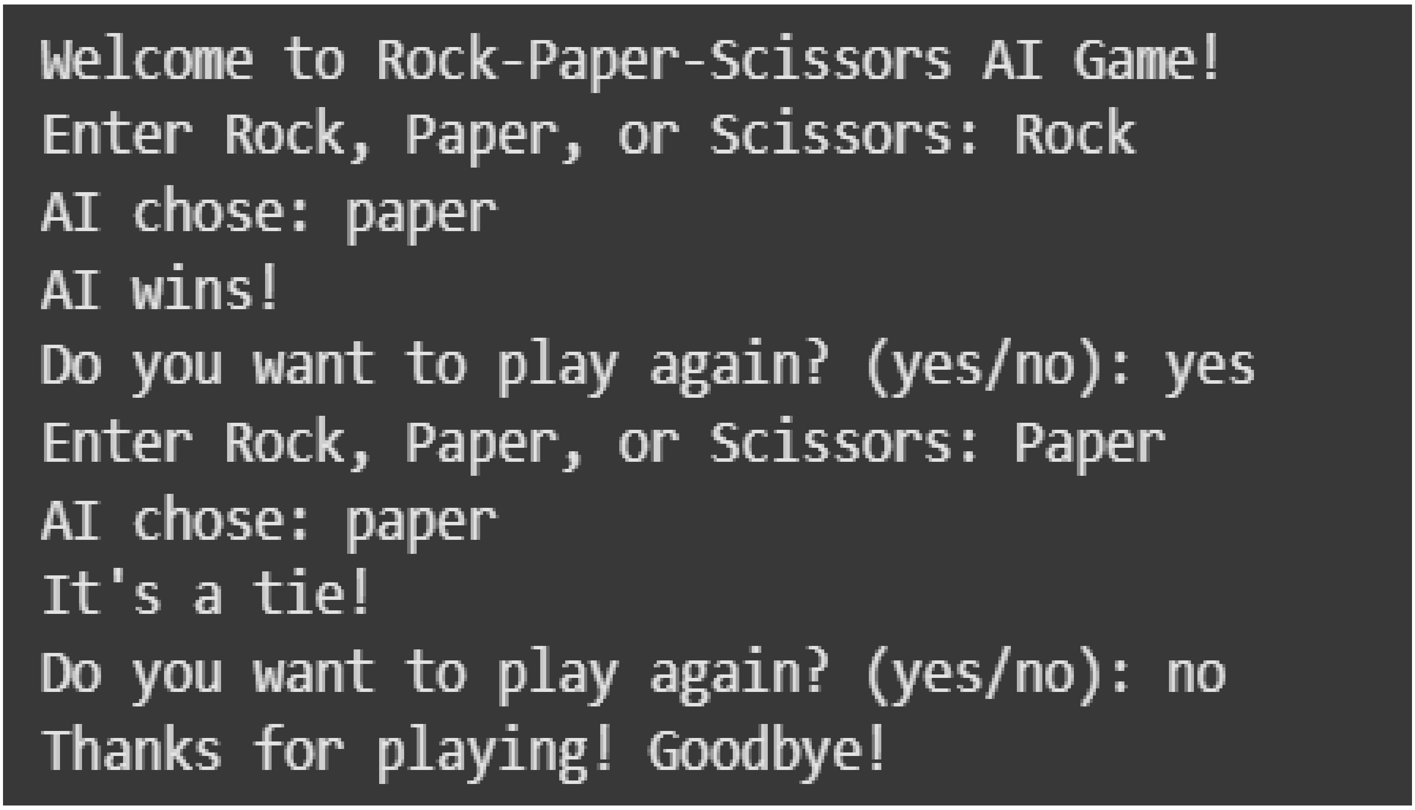
This approach ensures an engaging and interactive gameplay experience while allowing AI to adapt to the user's choices. 🚀

CODE:

```
import random

def get_user_choice():
    """
    Gets the user's choice and validates input.
    Ensures that the user enters 'rock', 'paper', or 'scissors' correctly.
    """
    choices = ["rock", "paper", "scissors"]
    user_choice = input("Enter Rock, Paper, or Scissors: ").strip().lower()

    while user_choice not in choices:
        print("Invalid choice! Please enter Rock, Paper, or Scissors.")
        user_choice = input("Enter Rock, Paper, or Scissors: ").strip().lower()

    return user_choice

def get_ai_choice(previous_user_choice=None):
    """
    AI selects a move. If a previous user move is available, AI uses a simple strategy:
    - It predicts that the user might repeat their previous move.
    - AI then chooses the best counter-move.
    - If no previous move exists, AI selects randomly.
    """
    choices = ["rock", "paper", "scissors"]

    # AI strategy: Counter the previous user choice if available
    if previous_user_choice:
        counter_moves = {"rock": "paper", "paper": "scissors", "scissors": "rock"}
        return counter_moves[previous_user_choice]

    return random.choice(choices)  # Default: Random choice

def determine_winner(user, ai):
    """
    Determines the winner based on the game rules.
    - Returns a message indicating if the user won, lost, or tied.
    """
    if user == ai:
        return "It's a tie!"
    elif (user == "rock" and ai == "scissors") or \
         (user == "scissors" and ai == "paper") or \
         (user == "paper" and ai == "rock"):
        return "You win!"
    else:
        return "AI wins!"

def play_game():
    """
    Main game loop:
    - Takes user input.
    - AI makes a decision.
    - Determines the winner.
    - Repeats until the user decides to quit.
    """
    print("Welcome to Rock-Paper-Scissors AI Game!")
    previous_user_choice = None  # Track user's last move for AI strategy

    while True:
        # Get user input
        user_choice = get_user_choice()

        # AI makes its move
        ai_choice = get_ai_choice(previous_user_choice)
        print(f"AI chose: {ai_choice}")

        # Determine winner
        result = determine_winner(user_choice, ai_choice)
        print(result)

        # Store previous user choice for AI strategy in next round
        previous_user_choice = user_choice

        # Ask if the user wants to play again
        play_again = input("Do you want to play again? (yes/no): ").strip().lower()
        if play_again != "yes":
            print("Thanks for playing! Goodbye!")
        break

# Run the game
play_game()
```

Output / Result

Below is an example of the expected output when the Rock-Paper-Scissors AI game is played:



```
Welcome to Rock-Paper-Scissors AI Game!
Enter Rock, Paper, or Scissors: Rock
AI chose: paper
AI wins!
Do you want to play again? (yes/no): yes
Enter Rock, Paper, or Scissors: Paper
AI chose: paper
It's a tie!
Do you want to play again? (yes/no): no
Thanks for playing! Goodbye!
```

References / Credits

Python Documentation – Used for understanding input handling and random number generation.

- https://docs.python.org/3/

Google Colab – Platform used for writing and executing the code.

- https://colab.research.google.com/

Rock-Paper-Scissors Game Logic – Basic rules and mechanics of the game.

- https://en.wikipedia.org/wiki/Rock_paper_scissors

GitHub – Repository for storing and sharing the project.

- https://github.com/