ROCK PAPER SCISSORS GAME

INTRODUCTION:

This Rock, Paper, Scissors game allows the user to play against the computer. The game follows the simple rules of rock beating scissors, scissors beating paper, and paper beating rock. The program will display the winner of each round, show the choices made by both the user and the computer, and keep track of the scores. The user will also have the option to play multiple rounds.

Requirements: Python 3.x

The random library for generating the computer’s choice.

Main Features:

1. **User Prompt**: The user is prompted to choose rock, paper, or scissors.

2. **Computer Selection**: The computer's choice is generated randomly from the options.

3. **Game Logic**: The program compares the user’s choice and the computer’s choice to determine the winner based on the rules:

* Rock beats Scissors.
* Scissors beat Paper.
* Paper beats Rock.

4. **Result Display**: The program will display both the user’s and the computer’s choices, and then declare the winner (either the user, the computer, or a tie).

5**. Score Tracking**: Optionally, scores for the user and computer can be tracked across multiple rounds.

6. **Play Again:** After each round, the user can choose to play again or exit.

Instructions:

1. The game will ask you to select one of the following choices:

**Rock**

**Paper**

**Scissors**

2. The computer will randomly choose one of these options as well.

3. The rules are simple:

* Rock beats Scissors.
* Scissors beat Paper.
* Paper beats Rock.

4. After each round, the results will be displayed, along with the current scores if multiple rounds are played.

5. You will be asked if you want to play another round after each game.

**Code Structure**

import random

# Function to determine the winner

def determine\_winner(user\_choice, computer\_choice):

if user\_choice == computer\_choice:

return "It's a tie!"

elif (user\_choice == "rock" and computer\_choice == "scissors") or \

(user\_choice == "scissors" and computer\_choice == "paper") or \

(user\_choice == "paper" and computer\_choice == "rock"):

return "You win!"

else:

return "You lose!"

# Function to get computer's choice

def get\_computer\_choice():

return random.choice(["rock", "paper", "scissors"])

# Main game loop

def play\_game():

user\_score = 0

computer\_score = 0

while True:

# Prompt user for choice

user\_choice = input("Enter 'rock', 'paper', or 'scissors': ").lower()

while user\_choice not in ["rock", "paper", "scissors"]:

user\_choice = input("Invalid input! Please enter 'rock', 'paper', or 'scissors': ").lower()

# Get computer's choice

computer\_choice = get\_computer\_choice()

# Display choices

print("Your choice: {user\_choice}")

print(“Computer's choice: {computer\_choice}")

# Determine the winner

result = determine\_winner(user\_choice, computer\_choice)

print(result)

# Update scores

if result == "You win!":

user\_score += 1

elif result == "You lose!":

computer\_score += 1

# Display score

Print("Score: You - {user\_score}, Computer - {computer\_score}")

# Ask user if they want to play again

play\_again = input("Do you want to play again? (yes/no): ").lower()

if play\_again != "yes":

print("Thanks for playing!")

break

# Start the game

play\_game()

Detailed Explanation of Code:

1. Imports:The random module is imported to randomly generate the computer’s choice.

2. Functions:\_winner(user\_choice, computer\_choice): This function compares the user’s and the computer’s choices, and returns the result ("You win!", "You lose!", or "It's a tie!").

get\_computer choice(): This function randomly picks one of the options ("rock", "paper", or "scissors") for the computer.

3. Main Game Loop:The play\_game() function manages the game, prompting the user for input, generating the computer's choice, determining the winner, updating the score, and asking if the user wants to play again.

4. User Interaction:

The program asks the user to input one of the three options: "rock", "paper", or "scissors". If the input is invalid, the user is prompted again.

After the round, the result and scores are displayed, and the user is asked if they want to play again.

**Optional Features:**

1. Score Tracking: The user's and computer’s scores are tracked throughout the game.

2. Play Again: After each round, the game asks if the user wants to play another round. If they choose "yes", the game continues; if they choose "no", the game ends.

**User Interface:**The user interface is text-based, with clear prompts and instructions.

The program asks for input and provides feedback after each round.

After each round, the scores are displayed, and the user is given the option to play again.

Conclusion:

This game provides a simple yet engaging experience where the user can compete against the computer in a classic game of Rock, Paper, Scissors. The program is interactive, keeping track of scores and offering the option to play again. The code is easy to follow and can be extended or modified for additional features or complexity.