Rock Paper, Scissors Final Project Report

Corey Merritt and Jackson Hay

Bellarmine University

CS-150

Dr. Sarkar

December 12, 2023

**Abstract:**

Our project is designed to recreate the historical game rock, paper, scissors in a interactive and innovative way. The game takes user input and compares it to the random choice by the computer and determines a winner. This report outlines the project's objectives, methodologies, outcomes, and the implementation of key features.

**Introduction:**

In a world were games are becoming more interactive and software developed we take things back to a simple time and create the classic game of Rock, Paper, Scissors but with this software developed innovation. The interactive game acts as a user friendly time wasting game. It implements several features in python to simplify and make the user’s experience more pleasuable.

**Methodology:**

We used python to develop this game and used many key features with the programming language such as classes, functions, and input statements to name a few. When we created the game we knew that we needed to find a way to keep track of the score and we used an if statement and developed a class to do so. We knew the key to this project would be the utilization of several classes to define and calculate everything.

**Design and Implementation:**

We used a ­­\_\_init\_\_ statement to give the options of rock, paper, scissors and to make a score tracker for bothe the user and CPU. Then used two classes so that a user can input their choice and for the CPU to generate a choice. We used an if statement within a class to compare the choices and to add a point to the winner’s score. This will repeat until the user types no when they are finished playing and then the game will end.

import random

class RPSGame:

def \_\_init\_\_(self):

self.choices = ["rock", "paper", "scissors"]

self.player\_score = 0

self.computer\_score = 0

def get\_player\_choice(self):

while True:

choice = input("Enter your choice (rock, paper, scissors): ").lower()

if choice in self.choices:

return choice

else:

print("Invalid choice. Please enter rock, paper, or scissors.")

def get\_computer\_choice(self):

return random.choice(self.choices)

def determine\_winner(self, player\_choice, computer\_choice):

if player\_choice == computer\_choice:

return "It's a tie!"

elif (

(player\_choice == "rock" and computer\_choice == "scissors")

or (player\_choice == "paper" and computer\_choice == "rock")

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

):

self.player\_score += 1

return "You win!"

else:

self.computer\_score += 1

return "Computer wins!"

def display\_scores(self):

print(f"Player Score: {self.player\_score} | Computer Score: {self.computer\_score}")

def play\_game(self):

print("Let's play Rock-Paper-Scissors!")

while True:

player\_choice = self.get\_player\_choice()

computer\_choice = self.get\_computer\_choice()

print(f"\nYou chose: {player\_choice}")

print(f"Computer chose: {computer\_choice}")

result = self.determine\_winner(player\_choice, computer\_choice)

print(result)

self.display\_scores()

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

if play\_again != "yes":

print("Thanks for playing!")

break

if \_\_name\_\_ == "\_\_main\_\_":

game = RPSGame()

game.play\_game()

**Results/Findings:**

As seen with the code above we used classes to set up every part of the game. We first start by inputing random so we input a random choice for the CPU later in the code. The first class uses self functions, the first to create a list of the choices Rock, Paper, Scisors, the second and third create a score tracker for the user and CPU. The second class uses a input statement to allow the user to make their choice and generates an ivalid message if user chooses an option not on the original list. The third class uses a random choice function to generate the CPU’s choice. The next class uses an if statement to compare the answers and to add a point to the winner. Then the class after uses a print statement to show the current score. The last class is used to show the results and to allow the user to end the game using a break.

**Discussion:**

The significance of this project is allow everyday users to play a interractive version of Rock, Paper, Scissors online. This is important because it get be an enjoyable for people to pass time and help anti-social play a simple but entertaining game. Things we could improve on is createing an ultiime score counter, an adaptable AI that makes smarter deciscions based on users previous choices and patterns to make the game more difficult, or add a way to adjust the CPU’s difficulty, and allow a user to face another user.

**Conclusion:**

We succesfully built the classic game Rock, Paper, Scissors to allow a user to face a computer. The code generates the user choice and compares it to the choice of the CPU and accurtely awards a point to the winner and continues to track this until the user decides to end the game and exit the code. We did this by using python to create functions, classes, if statements, and other important Python features. Overall, we created a highly interactive game of Rock, Paper, Scissors that is not only simple but enjoyable for the user.