INSTITUTE NAME: Besant Technologies

PROJECT: Rock Paper Scissor Game

SUBMITTED TO: Gowthami Shyam

SUBMITTED BY: Mohamed Rishwan A

CO-ORDINATOR NAME: Durika M

DATE OF SUBMISSION: 15-08-2025

TABLE OF CONTENT

ACKNOWLEDGEMENT

I take this opportunity to express my heartfelt gratitude to all those who supported and guided me throughout the successful completion of this project titled “Rock Paper Scissor Game”.

First and foremost, I would like to extend my sincere thanks to Ms. Gowthami Shyam, under whose guidance this project was completed. Her constant support, valuable feedback, and expert insights have helped me understand the concepts better and bring this project to completion with clarity and precision.

I would also like to express my special thanks to my project coordinator, Ms. Durika M, for her consistent encouragement, timely assistance, and technical inputs throughout the project. Her enthusiasm and constructive suggestions were extremely helpful during the development of this work.

I am deeply thankful to Besant Technologies, for providing me with the necessary platform and learning environment to work on this project. The institute’s resources and professional mentorship have played a vital role in enhancing my practical skills.

I would also like to thank my friends and family members for their motivation, moral support, and encouragement throughout this journey. Their presence has been a source of strength and positivity.

Finally, I extend my gratitude to everyone who directly or indirectly contributed to the successful execution of this project.

INTRODUCTION

The project titled “Rock Paper Scissor Game” is a simple yet engaging Python-based console game that demonstrates fundamental programming concepts such as conditional statements, loops, randomization, and user interaction. Inspired by the classic hand game "Rock, Paper, Scissors", this project aims to simulate a competitive game between the player and the computer, incorporating logic to determine the outcome of each round and maintaining a real-time scoreboard to enhance user engagement.

The primary objective of this project is to build a command-line game that randomly generates the computer's choice and compares it with the player's input to decide the winner. It provides the user with a continuous gaming experience until they choose to exit. It is an ideal project for beginners in programming as it introduces them to basic Python constructs like if-else conditions, loops, and list handling. The inclusion of the random module allows the computer to make unpredictable choices, making the game both fair and challenging.

Beyond entertainment, this project serves as a practical demonstration of core programming logic and helps users understand the importance of control flow, input validation, and user feedback in software applications. The game also maintains a scoreboard throughout the session, tracking the number of wins, losses, and ties, which adds to the sense of competitiveness. Overall, the "Rock Paper Scissor Game" is an effective combination of learning and fun, offering foundational programming practice while delivering a familiar and interactive user experience.

CONCEPTS USED IN PROJECT

The **"Rock Paper Scissor Game"** project utilizes several core concepts of Python programming. Though the game appears simple on the surface, it incorporates foundational programming principles that are essential for building more complex applications. This section outlines the key concepts implemented in this project and their roles in making the game functional and interactive.

1. **Input/Output Handling**:
   The project interacts with the user through the console using the input() function to get the player's choice and the print() function to display the result of each round, current scores, and final output. This demonstrates how a program can communicate with users effectively through standard input and output mechanisms.

2. **Conditional Statements (if-elif-else)**:
   Decision-making is at the heart of this project. The game logic is built using if-elif-else structures to determine the outcome of each round — whether the player wins, loses, or ties. These conditional checks compare the player's input and the computer's choice to apply game rules accordingly.

3. **Loops (while loop)**:
   The entire game runs inside an infinite while loop that continues until the user decides to exit. This loop ensures that the game is playable in multiple rounds, keeping track of the score and providing a seamless experience.

4. **Randomization (random module)**:
   To make the computer's choice unpredictable, the project uses Python's random module — specifically the choice() function — to randomly select between "rock", "paper", and "scissor". This introduces the concept of pseudo-random behavior in programming and adds fairness to the gameplay.

5. **List Data Structure**:
   The available choices ("rock", "paper", "scissor") are stored in a list, showcasing how collections are used to manage a set of related data. Lists are also used in conjunction with the random.choice() function to access random elements.

6. **Score Tracking Using Variables**:
   Three separate variables are maintained to keep count of the player's wins, computer's wins, and tie matches. This highlights the concept of state management using variables and how values can be updated during the execution of a program.

7. **String Manipulation and Validation**:
   The project converts all player inputs to lowercase using .lower() to ensure consistent comparison, regardless of how the user types their choice. It also checks whether the input is valid, teaching input validation techniques.

Through these concepts, the project not only delivers an enjoyable gaming experience but also strengthens the understanding of Python basics. It serves as a practical example for beginners to see how simple logic, when combined properly, can lead to a fully functional interactive program.

SOURCE CODE

```
import random as r
choices = ["rock", "paper", "scissor"]
player_score = 0
computer_score = 0
tie_score = 0
print("ROCK PAPER SCISSOR GAME")
print("Type 'exit' to quit\n")
while True:
    player_choice = input("Enter your choice:").lower()
    if player_choice == "exit":
        print("\nThanks for playing!")
        print("======== FINAL SCOREBOARD ========")
        print(f"You: {player_score}")
        print(f"Computer: {computer_score}")
        print(f"Ties: {tie_score}")
        print("==================================")
        break
    if player_choice not in choices:
        print("Invalid choice! Try again.\n")
        continue
    computer_choice = r.choice(choices)
    print(f"Computer chose: {computer_choice}")
    if player_choice == computer_choice:
        print("It's a tie!\n")
        tie_score += 1
    elif (player_choice == "rock" and computer_choice == "scissor") or \
         (player_choice == "paper" and computer_choice == "rock") or \
         (player_choice == "scissor" and computer_choice == "paper"):
        print("You win!\n")
        player_score += 1
    else:
        print("Computer wins!\n")
        computer_score += 1
```

WORKING PROCEDURE OF SOURCE CODE

The Rock Paper Scissor Game developed in Python works on a continuous loop, allowing users to repeatedly play rounds against a computer opponent. The source code follows a systematic procedure for taking input, processing the logic, and displaying the results. Here's a step-by-step breakdown of the working mechanism of the program.

1. **Initialization and Setup:**
   The code begins by importing the random module to simulate the computer's choice. A list named choices stores the three possible options: "rock", "paper", and "scissor". Three score variables (player_score, computer_score, and tie_score) are initialized to 0 to keep track of the gameplay results.

2. **Welcome Message and Instructions:**
   The game displays a welcome message and instructions, informing the user that they can type "exit" to quit the game at any point.

3. **Main Game Loop (While Loop):**
   The main logic is enclosed inside a while True loop. This allows the game to continuously prompt the user to enter their choice until they type "exit".

4. **Player Input and Validation:**
   The user's input is taken using input() and immediately converted to lowercase using .lower() to avoid case sensitivity issues. If the player types "exit", the loop breaks, and the final scores are displayed. If the input is not one of the valid choices, the game displays an "Invalid choice" message and re-prompts for input.

5. **Computer Choice:**
   Using random.choice(choices), the computer randomly selects either "rock", "paper", or "scissor". This adds unpredictability and fairness to the game.

6. **Result Evaluation Using Conditions:**
   The player's choice is compared with the computer's choice using if-elif-else conditions:
   - If both choices are the same, the result is a tie, and the tie_score is incremented.

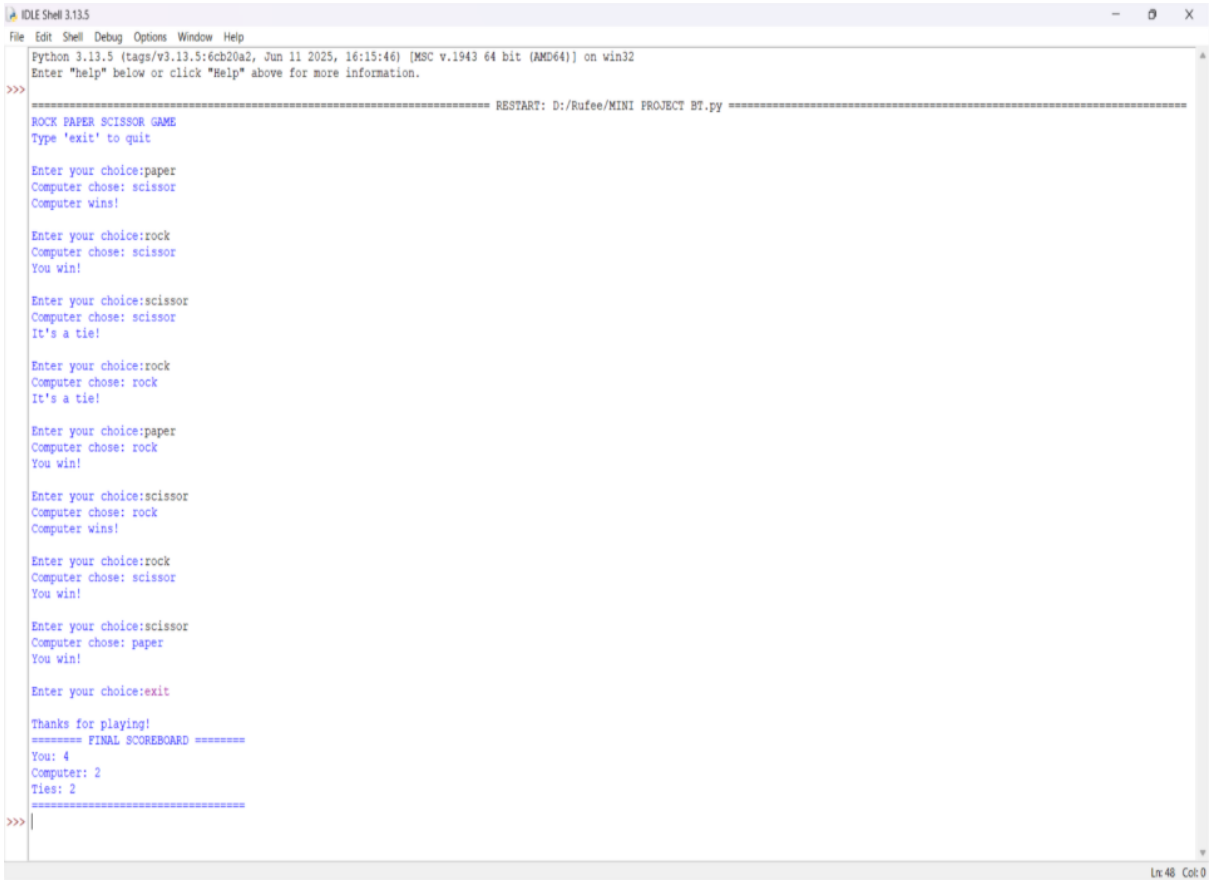
- If the player wins according to game rules (rock beats scissor, paper beats rock, scissor beats paper), player_score is incremented.
- Otherwise, the computer_score is incremented.

7. **Result Display:**
   After each round, the result of that round (win, loss, or tie) is displayed along with the computer's choice.

8. **Final Scoreboard:**
   Once the player exits the game, the final scoreboard is printed, showing the total number of player wins, computer wins, and tie games. This summary helps the player see how they performed overall.

This step-by-step structure ensures the code is not only easy to understand but also demonstrates how logic and user interaction work together in Python programming. The project is ideal for beginners to practice control flow, randomization, and scorekeeping using simple constructs.

OUTPUT


IDLE Shell 3.13.5
File Edit Shell Debug Options Window Help
Python 3.13.5 (tags/v3.13.5:6cb20a2, Jun 11 2025, 16:15:46) [MSC v.1943 64 bit (AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
========== RESTART: D:/Rufee/MINI PROJECT BT.py ==========
ROCK PAPER SCISSOR GAME
Type 'exit' to quit
Enter your choice:paper
Computer chose: scissor
Computer wins!
Enter your choice:rock
Computer chose: scissor
You win!
Enter your choice:scissor
Computer chose: scissor
It's a tie!
Enter your choice:rock
Computer chose: rock
It's a tie!
Enter your choice:paper
Computer chose: rock
You win!
Enter your choice:scissor
Computer chose: rock
Computer wins!
Enter your choice:rock
Computer chose: scissor
You win!
Enter your choice:scissor
Computer chose: paper
You win!
Enter your choice:exit
Thanks for playing!
======== FINAL SCOREBOARD ========
You: 4
Computer: 2
Ties: 2
==========
>>>
Ln: 48 Col: 0

CONCLUSION

The Rock Paper Scissor Game project serves as an excellent example of how fundamental programming concepts can be applied to create interactive and entertaining applications using Python. This project demonstrates the practical implementation of conditional statements, loops, user input handling, and randomization. The simplicity of the game makes it ideal for beginners, yet its structure offers meaningful insight into how real-time applications operate in response to user actions.

Through this project, we successfully built a fully functional game that allows a user to compete with the computer in a fair and randomized manner. The inclusion of a score-tracking mechanism adds an element of engagement and competitiveness. The use of a loop ensures that the game continues until the user explicitly exits, allowing repeated gameplay without restarting the program. Error handling is also incorporated to manage invalid inputs, thus making the game user-friendly and reliable.

In conclusion, the project has not only enhanced our programming skills but has also deepened our understanding of logic building, flow control, and code modularity. It highlights the importance of clear logic structure and user experience design in software development. Moving forward, the project can be expanded with additional features like a GUI (Graphical User Interface), difficulty levels, or multiplayer support. This foundational project has laid the groundwork for more complex Python-based applications and serves as a stepping stone toward becoming proficient in software development.

BIBLIOGRAPHY

1. Besant Technologies – Course Materials and Live Sessions – Provided foundational knowledge and support in Python programming and logic-building during the course.
2. Guidance by Ms. Gowthami Shyam (Trainer) and Co-ordinator Ms. Durika M, *Besant Technologies* – Their instruction and mentorship played a significant role in successfully completing this project.

GITHUB AND LINKEDIN LINKS

1. https://github.com/Mohamed-Rishwan/Rock-Paper-Scissor
2. http://www.linkedin.com/in/mohamed-rishwan-a-284408375