Rock, Paper, Scissors

Project Description

This project consists of creating a Rock, Paper, Scissors game by using Python. The user will be prompted to input one of three options: rock, paper, or scissors. The computer will choose one option at random. Then, both choices will be compared to determine who wins the game.

Languages and Utilities used

Python

Visual Studio Code

Program walk-through:

Import the module

```
1    import random
```

Before starting, I imported the random module to use different tools inside it to randomize computer's actions the game.

Take user input

```
user_action = input("Enter a choice (rock, paper, scissors): ")
```

This will prompt the user to enter a selection and save it to the user_action variable for later use. Now that the user has selected an action, the computer needs to decide what to do.

Make computer choose

```python
possible_actions = ["rock", "paper", "scissors"]
computer_action = random.choice(possible_actions)
```

A list containing all three possible choices is created and stored in the possible_actions variable. Then, a random element is selected from the list by using the random.choice() method, which will be stored in the computer_action variable.

Determine a winner

```python
if user_action == computer_action:
    print(f"Both players selected {user_action}. It's a tie!")
elif user_action == "rock":
    if computer_action == "scissors":
        print("Rock smashes scissors! You win!")
    else:
        print("Paper covers rock! You lose.")
elif user_action == "paper":
    if computer_action == "rock":
        print("Paper covers rock! You win!")
    else:
        print("Scissors cuts paper! You lose.")
elif user_action == "scissors":
    if computer_action == "paper":
        print("Scissors cuts paper! You win!")
    else:
        print("Rock smashes scissors! You lose.")
```

Using an if, elif, and else block, I can compare players' choices and determine a winner. By comparing the tie condition first, I get rid of quite a few cases. If I didn't do that, then I would've needed to check each possible action for user_action and compare it against each possible action for computer_action. By checking the tie condition first, I'm able to know what the computer chose with only two conditional checks of computer_action. A corresponding message is displayed depending on the results.