Developers Institute

Python Course

Mini-project: Rock, Paper, Scissors

Introduction

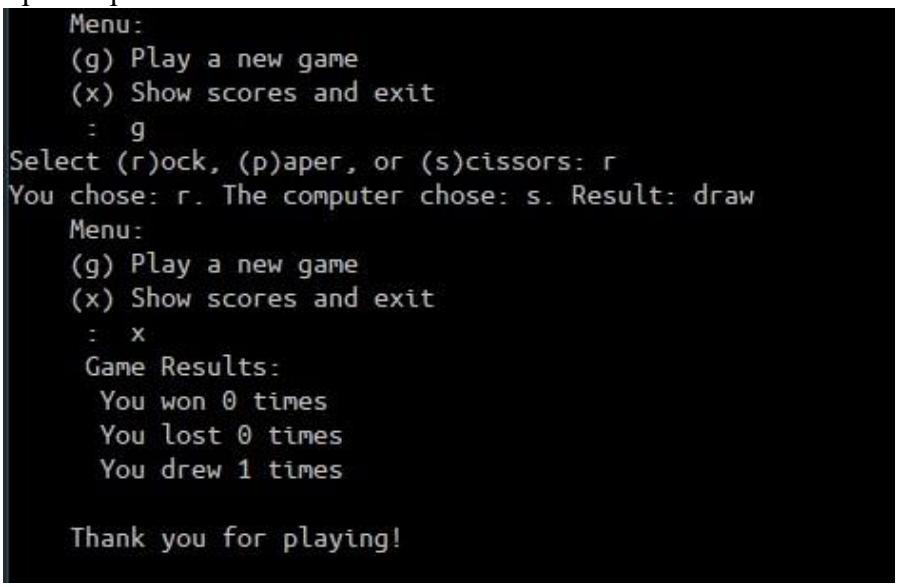
We will create a game for the user to play Rock-paper-scissors against the computer. The user will input his/her move (rock/paper/scissors) on the keyboard, and the computer will select either rock, paper, or scissors at **random**. The computer will then compare the user's move with the computer's move, and determine the results of the game:

1. The user won,
2. The computer won (the user lost), or
3. A draw (tie)

We will print the outcome of each game: the user's choice, the computer's choice, and the result.

The user will be able to play again and again. Once the user decides to exit the program, we will print a summary of the outcomes of all the games: how many times he won, lost, and drew with the computer.

Here's some example output:



Instructions

1. Create a new directory for the game. Inside it, create 2 files:
   - **rock-paper-scissors.py** – this will contain functions to show the main menu, handle user's input, and show the game summary before exiting
   - **game.py** – this will contain a Game **class** which will have functions to play a single game of rock-paper-scissors against the computer, determine the game's result, and return the result.
2. **rock-paper-scissors.py -** this file should have 3 functions:

1. **get_user_menu_choice()** - this should display a simple menu, get the user's choice (with data validation), and return the choice. No looping should occur here.
2. **print_results(results)** – this should print the results of the games played. It should have a single parameter named '**results**', which will be a **dictionary** of the results of the games played. It should display these results in a user-friendly way, and thank the user for playing.
    - '**results**' should be in this form: { 0: 'win', 1: 'loss', 2: 'loss', 3: 'draw'}.
    - Bear in mind that this dictionary will need to be created and populated in some other part of our code, and passed in to the **print_results** function at the right time.

3. **main()** - the main function. It should take care of 3 things:
    1. displaying the menu repeatedly, until the user types in the value to exit the program, eg. 'x' or 'q', whatever you decide. (Make use of the **get_user_menu_choice** function)
    2. When the user chooses to play a game:
        1. Create a new Game object (see below), and call its **play()** function, receiving the result of the game that is returned.
        2. Remember the results of every game that is played. More about this below.
        3. When the user chooses to exit the program, call the **print_results** function in order to display a summary of all the games played.

3. **game.py** – this file/module should contain a class called **Game**. It should have 4 functions:
    1. **get_user_item(self)** – Ask the user to select an item (rock/paper/scissors). Keep asking until the user has selected one of the items – use data validation and looping. **Return** the item at the end of the function.
    2. **get_computer_item(self)** – Select rock/paper/scissors **at random** for the computer. **Return** the item at the end of the function.
    3. **get_game_result(self, user_item, computer_item)** – Determine the result of the game.
        1. Parameters:
            1. **user_item** – the user's chosen item (rock/paper/scissors)
            2. **computer_item** – the computer's 'chosen' (random) item (rock/paper/scissors)
        2. **Return** either 'win', 'draw', or 'loss'.
    4. **play(self)** – the function that will be called from outside the class (ie. from rock-paperscissors.py). It will do 3 things:
        1. Get the user's item (rock/paper/scissors) and remember it
        2. Get a random item for the computer (rock/paper/scissors) and remember it
        3. Determine the results of the game by comparing the user's item and the computer's item
        4. **Print** the output of the game; something like this: "*You selected rock. The computer selected paper. You lose :(*", or "*You selected scissors. The computer selected scissors. You drew!*"
        5. **Return** the results of the game as a string: 'win'/'draw'/'loss', where 'win' means that the user has won, 'draw' means the user and the computer got the same item, and 'loss' means that the user has lost.