BAHRIA UNIVERSITY, Karachi Campus)

Department of Software Engineering

ASSIGNMENT # 03 – Fall 2022

Course Title:	**Software Construction**	Course Code:	**SEC-311**
Class:	**BSE – 5(A)**	Shift:	**Morning**
Course Instructor:	**Dr. Salahuddin Shaikh**	Date:	**14th Dec 2023**
Due Date:	**21st December 2023**	Max. Marks:	**5 Marks**

Question No. 1: **[CLO#04, 5 marks]**

Assignment Task: Refactoring Mini-Game Code

Objective:

Identify, analyze, and apply refactoring techniques to improve the structure and readability of the provided mini-game code.

Task Description:

You've been given a code snippet representing a simple mini-game. Your task is to review and refactor this code, applying appropriate refactoring techniques to enhance its readability, maintainability, and potentially its extensibility.

Code Snippet (Mini-Game):

```
import random

def play_game():
    player_score = 0
    computer_score = 0
    rounds = 3
    for _ in range(rounds):
        player_choice = input("Enter your choice (rock/paper/scissors): ")
        computer_choice = random.choice(['rock', 'paper', 'scissors'])
```

```
        if player_choice == computer_choice:
            print("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'):
            print("You win!")
            player_score += 1
        else:
            print("Computer wins!")
            computer_score += 1
    print("Final Scores:")
    print(f"Player: {player_score}")
    print(f"Computer: {computer_score}")
    if player_score > computer_score:
        print("Congratulations! You won the game!")
    elif player_score < computer_score:
        print("Sorry, you lost. Better luck next time!")
    else:
        print("It's a tie!")
play_game()
```

Instructions:

Analyze the given mini-game code snippet and identify areas for improvement in terms of readability, structure, or extensibility.

Apply refactoring techniques to improve the code's structure, readability, and potentially add functionality. Consider techniques like:

Extract Method

Simplify Conditional Logic

Use Constants or Enums

Encapsulate Game Logic into Functions or Classes

Improve Input Validation or Error Handling

Provide comments or explanations for the changes made, explaining how the refactored code improves upon the original version.

Optionally, consider expanding the game's functionality (e.g., adding more choices, improving the scoring system) while refactoring the code.

Optionally, run test cases or examples to demonstrate the correctness and functionality of the refactored code.

Deliverables:
Refactored mini-game code with comments explaining the applied refactoring techniques and their benefits.
A brief explanation or summary detailing the improvements made and how they enhance the code.
Optional: Additional functionality or enhancements made to the game (if any).
Optional: Test cases or examples showcasing the functionality of the refactored code.

Submission Format:

Submit a document or code file containing the refactored mini-game code, comments, explanations, and any additional enhancements made.