9.8-Introduction to Stack & Queue

Program

```
using System;

class GuessingGame
{
    public void Run()
    {
        // Generate a random number between 1 and 100
        Random random = new Random();
        int randomNumber = random.Next(1, 101);

        // Initialize variables
        int guess;
        int attempts = 0;

        Console.WriteLine("Welcome to the Guessing Game!");
        Console.WriteLine("Guess a number between 1 and 100.");

        // Allow the player to make guesses until they guess the correct number
        do
        {
            Console.Write("Enter your guess: ");
            guess = int.Parse(Console.ReadLine());
            attempts++;
            if (guess < randomNumber)

            {

                Console.WriteLine("Too low! Try again.");

            }

            else if (guess > randomNumber)

            {

                Console.WriteLine("Too high! Try again.");

            }

            else

            {

                Console.WriteLine($"Congratulations! You guessed the number {randomNumber} in {attempts} attempts.");

            }

        } while (guess != randomNumber);

    }


    static void Main()

    {

        GuessingGame game = new GuessingGame();

        game.Run();

    }

}
```

Exercise

Write a C# program that develop a console-based guessing game in C#.

The program generates a random number between 1 and 100.

It prompts the player to guess the number.

After each guess, it provides feedback indicating whether the guess is too low, too high, or correct.

It continues to prompt the player for guesses until they correctly guess the randomly generated number.

Random number generation using Random class.

User input handling to accept guesses.

Feedback messages based on the comparison between the guess and the random number.

Tracking the number of attempts made by the player.

Displaying a congratulatory message upon guessing the correct number, along with the number of attempts.

Utilizes a do-while loop to repeatedly prompt the player for guesses until the correct number is guessed.

Uses Random.Next() method to generate a random number within the specified range.

Tracks the number of attempts using a counter variable.

Provides feedback to the player based on their guesses using conditional statements.

The Main() method initializes an instance of the GuessingGame class and calls its Run() method to start the game.

User Interaction:

Displays welcoming messages and instructions to the player.

Accepts user input for guesses using Console.ReadLine().

Outputs feedback messages and congratulatory message using Console.WriteLine().

The game terminates once the correct number is guessed, ending the do-while loop and displaying the congratulatory message.

Hint

Random Number Generation: Utilize the Random class to generate a random number between 1 and 100.

User Input Handling: Use Console.ReadLine() to accept user input for their guesses.

Looping Structure: Implement a do-while loop to repeatedly prompt the player for guesses until they guess the correct number.

Feedback Messages: Provide feedback to the player based on their guesses using Console.WriteLine().

Tracking Attempts: Keep track of the number of attempts made by the player using a counter variable.

Conditional Statements: Utilize if-else statements to compare the guess with the random number and provide appropriate feedback.

Termination Condition: Define a condition for terminating the loop when the correct number is guessed.

Congratulatory Message: Display a congratulatory message along with the number of attempts upon guessing the correct number.

Explanation

The Guessing Game program begins by generating a random number between 1 and 100 using the Random class. It then prompts the user to guess this number and initializes variables to track the user's guess and the number of attempts made. The program enters a do-while loop, allowing the player to make guesses until they guess the correct number.

Within the loop, the program reads the user's guess from the console and increments the attempts counter. It compares the guess with the randomly generated number and provides feedback to the player, indicating whether the guess is too low, too high, or correct. If the guess is correct, the program displays a congratulatory message along with the number of attempts taken.

The loop continues until the user correctly guesses the random number, at which point the program terminates. Overall, the Guessing Game program provides an interactive experience for users to test their guessing skills while providing feedback on their progress until they successfully guess the target number.