Guessing the Number Game

Submitted by:
Vishesh Agarwal

Date:
10-03-2025

---

2. Introduction

The **Guessing the Number Game** is a simple interactive console game where the user has to guess a randomly generated number within a user defined range. The game presents a series of challenges that help players practice problem-solving and critical thinking while enjoying the process of guessing a number. This project is designed to demonstrate the application of basic programming concepts such as loops, conditional statements, user input handling, and exception handling in Python.

In this game, the user is prompted to enter a lower and upper limit to define the range for the random number. The player has five attempts to guess the number correctly. After each guess, the program provides feedback, indicating whether the guess was too high or too low, and the game ends when the user either guesses correctly or exhausts all attempts.

3. Methodology

The methodology for the development of this game is based on the use of Python programming language constructs such as loops, conditional statements, exception handling, and input/output functions. The following steps outline the process:

1. **Importing Required Modules**: The random module is imported to generate a random number in the specified range. The random.randint() function is used to generate an integer value within the range provided by the user.

2. **User Input**: The program prompts the user to input a lower and upper limit for the range in which the random number will be generated. This user input is then stored as integer values that will guide the number generation.

3. **Generating a Random Number**: Using the random.randint(a, b) function, the program generates a random integer between the two user-specified values a (lower limit) and b (upper limit).

4. **Game Loop**: A while loop runs five times (the allowed number of attempts). During each iteration of the loop, the program prompts the user for a guess. If the guess is correct, the program congratulates the user and ends. If the guess is incorrect, the program provides feedback ("Too low!" or "Too high!") to guide the user toward the correct answer.

5. **Exception Handling**: A try/except block ensures that the user inputs a valid integer. If the input is invalid (e.g., a non-numeric value), the program handles the exception and prompts the user to enter a valid number.

6. **Ending the Game**: If the user guesses the number correctly, the game congratulates the player and ends. If the user runs out of attempts without guessing the number, the game reveals the correct number and ends.

This simple approach allows the game to be played interactively in the console, providing an enjoyable experience while demonstrating basic concepts of Python programming.

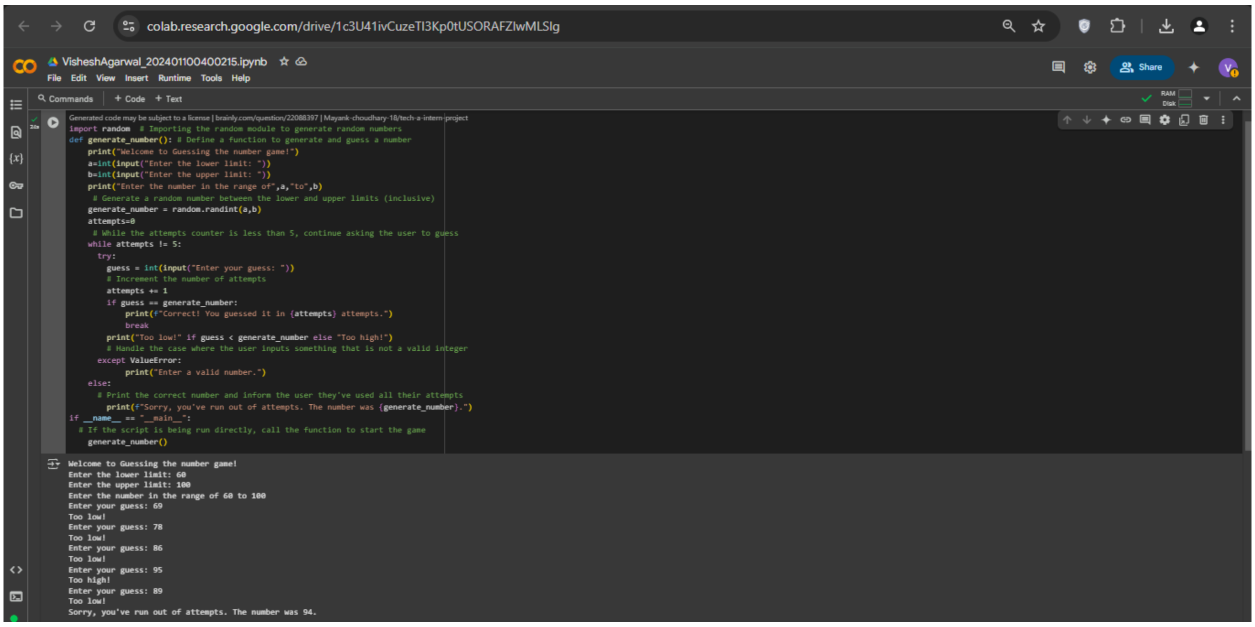
4. Code Typed

```
import random  # Importing the random module to generate random numbers
# Define a function to generate and guess a number
def generate_number():
    # Print a welcome message to the player
    print("Welcome to Guessing the number game!")
    # Ask the user to input the lower limit for the range
    a = int(input("Enter the lower limit: "))
    # Ask the user to input the upper limit for the range
    b = int(input("Enter the upper limit: "))
    # Print a message showing the range within which the user has to guess
    print("Enter the number in the range of", a, "to", b)
    # Generate a random number between the lower and upper limits (inclusive)
    generate_number = random.randint(a, b)
    # Initialize the number of attempts to 0
    attempts = 0
    # While the attempts counter is less than 5, continue asking the user to guess
    while attempts != 5:
        try:  # Ask the user to input their guess
            guess = int(input("Enter your guess: "))
            # Increment the number of attempts
            attempts += 1
            # If the guess is correct, print a success message and break the loop
            if guess == generate_number:
                print(f"Correct! You guessed it in {attempts} attempts.")
                break
            # If the guess is lower than the random number, give a hint
            print("Too low!" if guess < generate_number else "Too high!")
        # Handle the case where the user inputs something that is not a valid integer
        except ValueError:
            print("Enter a valid number.")
    # If the user runs out of attempts (i.e., the loop completes without breaking)
    else:
        # Print the correct number and inform the user they've used all their attempts
        print(f"Sorry, you've run out of attempts. The number was {generate_number}.")
# The following checks if the script is being run directly (not imported as a module)
if __name__ == "__main__":
    # If the script is being run directly, call the function to start the game
    generate_number()
```

Conclusion

The **Guessing the Number Game** is a simple yet engaging game that allows users to test their guessing skills. It uses basic Python constructs such as loops, conditionals, and exception handling to provide a smooth gaming experience. By generating a random number within a user-defined range and providing feedback based on the player's guesses, the game offers an interactive learning experience in programming. The project effectively demonstrates how to handle user input, generate random numbers, and manage game logic in Python.

Output Screenshot


colab.research.google.com/drive/1c3U41ivCuzeTI3Kp0tUSORAFZIwMLSIg
VisheshAgarwal_202401100400215.ipynb
File Edit View Insert Runtime Tools Help
Commands
+ Code
+ Text
Share
Generated code may be subject to a license | brainly.com/question/22088397 | Mayank-choudhary-18/tech-a-intern-project

```python
import random  # Importing the random module to generate random numbers
def generate_number(): # Define a function to generate and guess a number
    print("Welcome to Guessing the number game!")
    a=int(input("Enter the lower limit: "))
    b=int(input("Enter the upper limit: "))
    print("Enter the number in the range of",a,"to",b)
     # Generate a random number between the lower and upper limits (inclusive)
    generate_number = random.randint(a,b)
    attempts=0
     # While the attempts counter is less than 5, continue asking the user to guess
    while attempts != 5:
      try:
        guess = int(input("Enter your guess: "))
        # Increment the number of attempts
        attempts += 1
        if guess == generate_number:
            print(f"Correct! You guessed it in {attempts} attempts.")
            break
        print("Too low!" if guess < generate_number else "Too high!")
        # Handle the case where the user inputs something that is not a valid integer
      except ValueError:
            print("Enter a valid number.")
    else:
      # Print the correct number and inform the user they've used all their attempts
        print(f"Sorry, you've run out of attempts. The number was {generate_number}.")
if __name__ == "__main__":
  # If the script is being run directly, call the function to start the game
    generate_number()
```

```
Welcome to Guessing the number game!
Enter the lower limit: 60
Enter the upper limit: 100
Enter the number in the range of 60 to 100
Enter your guess: 69
Too low!
Enter your guess: 78
Too low!
Enter your guess: 86
Too low!
Enter your guess: 95
Too high!
Enter your guess: 89
Too low!
Sorry, you've run out of attempts. The number was 94.
```