MINIOR PROJECT

Submitted By : Akula Hema Venkata Sriram

Case Study: A Number Guessing Game

Introduction:

This case study examines a Python-based number guessing game designed to challenge users across different difficulty levels. The game is structured around selecting a random number within a specified range based on the chosen difficulty. Users are then prompted to guess the number within a limited number of attempts, with feedback provided after each guess. The objective is to enhance problem-solving and logical thinking, while adding a layer of entertainment through scoring and difficulty options.

Game Design:

The number guessing game consists of several core functions, each responsible for a specific aspect of the game mechanics. The game flow involves selecting a difficulty level, allowing users to guess the number, calculating a score based on performance, and prompting for replay. The functions are as follows:

1. choose_difficulty():

This function allows users to select a difficulty level:

- Easy: Numbers between 1 and 10, with 5 attempts.
- Medium: Numbers between 1 and 50, with 7 attempts.
- Hard: Numbers between 1 and 100, with 10 attempts.

Each difficulty level influences the range of numbers and the number of attempts the user is granted.

2. calculate_score():

The score calculation is based on the difficulty level and the number of attempts used. A higher difficulty level results in a higher base score. The game also rewards players for unused attempts.

3. play_guessing_game():

This is the core function where the actual gameplay occurs. The game selects a random number based on the chosen difficulty level, and the user is prompted to guess the number. The function provides feedback on each guess, informing the user whether their guess was too high, too low, or correct.

4. ask_to_play_again():

After each game session, users are asked whether they would like to play again. This function ensures that the game can be played repeatedly until the user decides to quit.

5. start_game():

The main driver of the game, this function handles the overall flow of the gameplay, calling the guessing game and replay prompts in a loop.

Game Mechanics:

Difficulty Selection:

The game provides three levels of difficulty, affecting the range of numbers and the number of attempts available. This ensures players of varying skill levels can enjoy the game.

Feedback on Guesses:

Users receive immediate feedback on whether their guess is too high or too low, allowing them to adjust subsequent guesses.

Scoring System:

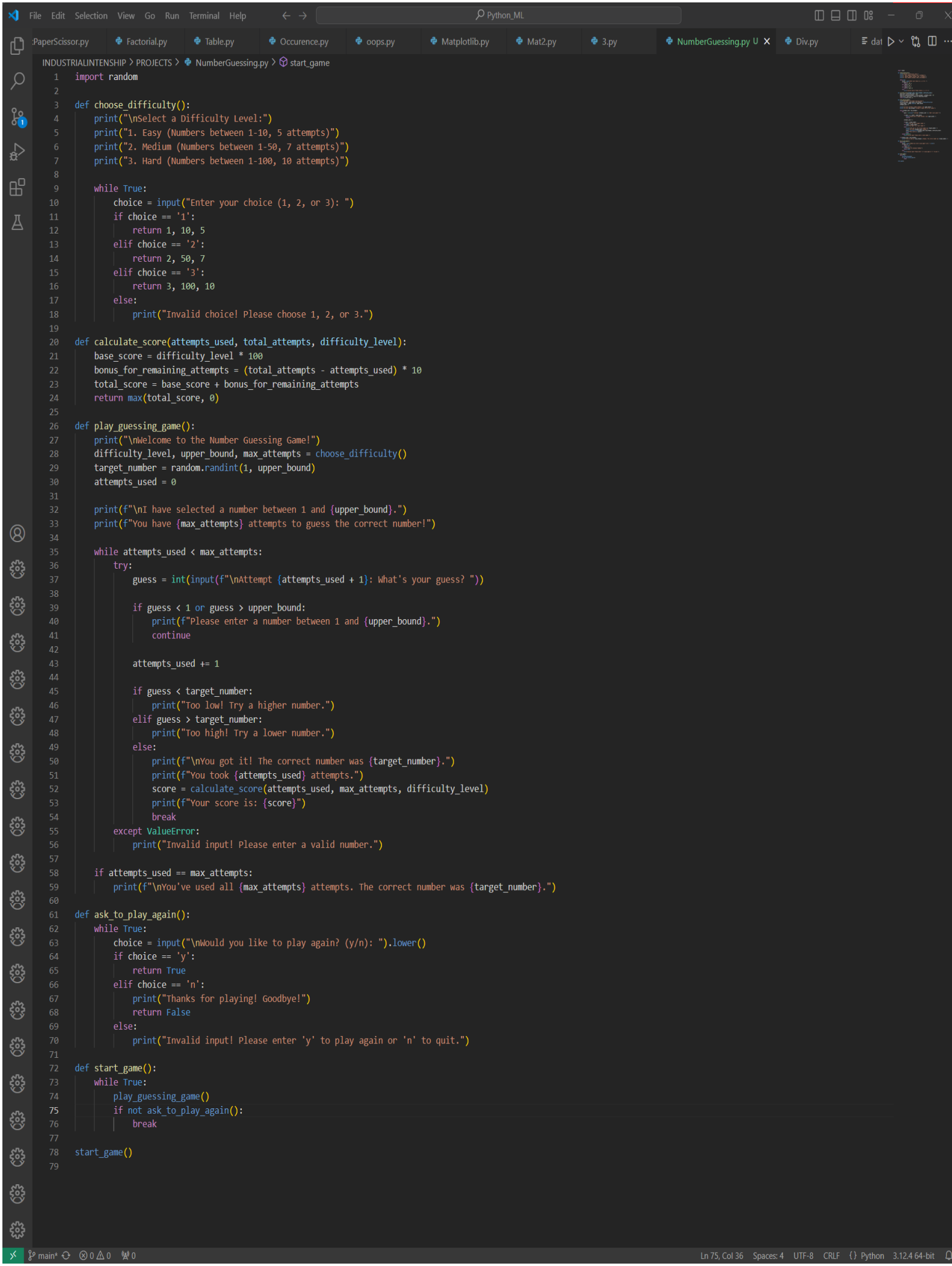
The scoring system incentivizes efficiency by rewarding unused attempts. The higher the difficulty and fewer attempts used, the higher the final score.

Replayability :

Players can choose to play multiple rounds, maintaining engagement and offering continuous gameplay.

Conclusion:

This number guessing game is a simple, interactive project that demonstrates effective use of Python for game development. The modular design of the game—separating difficulty selection, gameplay, score calculation, and replay functionality—ensures maintainability and scalability. This case study highlights how basic game mechanics can be implemented to create an enjoyable user experience.


File Edit Selection View Go Run Terminal Help
Python_ML
PaperScissor.py
Factorial.py
Table.py
Occurence.py
oops.py
Matplotlib.py
Mat2.py
3.py
NumberGuessing.py U
Div.py
INDUSTRIALINTENSHIP > PROJECTS > NumberGuessing.py > start_game
import random
def choose_difficulty():
print("\nSelect a Difficulty Level:")
print("1. Easy (Numbers between 1-10, 5 attempts)")
print("2. Medium (Numbers between 1-50, 7 attempts)")
print("3. Hard (Numbers between 1-100, 10 attempts)")
while True:
choice = input("Enter your choice (1, 2, or 3): ")
if choice == '1':
return 1, 10, 5
elif choice == '2':
return 2, 50, 7
elif choice == '3':
return 3, 100, 10
else:
print("Invalid choice! Please choose 1, 2, or 3.")
def calculate_score(attempts_used, total_attempts, difficulty_level):
base_score = difficulty_level * 100
bonus_for_remaining_attempts = (total_attempts - attempts_used) * 10
total_score = base_score + bonus_for_remaining_attempts
return max(total_score, 0)
def play_guessing_game():
print("\nWelcome to the Number Guessing Game!")
difficulty_level, upper_bound, max_attempts = choose_difficulty()
target_number = random.randint(1, upper_bound)
attempts_used = 0
print(f"\nI have selected a number between 1 and {upper_bound}.")
print(f"You have {max_attempts} attempts to guess the correct number!")
while attempts_used < max_attempts:
try:
guess = int(input(f"\nAttempt {attempts_used + 1}: What's your guess? "))
if guess < 1 or guess > upper_bound:
print(f"Please enter a number between 1 and {upper_bound}.")
continue
attempts_used += 1
if guess < target_number:
print("Too low! Try a higher number.")
elif guess > target_number:
print("Too high! Try a lower number.")
else:
print(f"\nYou got it! The correct number was {target_number}.")
print(f"You took {attempts_used} attempts.")
score = calculate_score(attempts_used, max_attempts, difficulty_level)
print(f"Your score is: {score}")
break
except ValueError:
print("Invalid input! Please enter a valid number.")
if attempts_used == max_attempts:
print(f"\nYou've used all {max_attempts} attempts. The correct number was {target_number}.")
def ask_to_play_again():
while True:
choice = input("\nWould you like to play again? (y/n): ").lower()
if choice == 'y':
return True
elif choice == 'n':
print("Thanks for playing! Goodbye!")
return False
else:
print("Invalid input! Please enter 'y' to play again or 'n' to quit.")
def start_game():
while True:
play_guessing_game()
if not ask_to_play_again():
break
start_game()
main*
Ln 75, Col 36
Spaces: 4
UTF-8
CRLF
Python
3.12.4 64-bit

```
Welcome to the Number Guessing Game!

Select a Difficulty Level:
1. Easy (Numbers between 1-10, 5 attempts)
2. Medium (Numbers between 1-50, 7 attempts)
3. Hard (Numbers between 1-100, 10 attempts)
Enter your choice (1, 2, or 3): 3

I have selected a number between 1 and 100.
You have 10 attempts to guess the correct number!

Attempt 1: What's your guess? 50
Too high! Try a lower number.

Attempt 2: What's your guess? 25
Too high! Try a lower number.

Attempt 3: What's your guess? 15
Too high! Try a lower number.

Attempt 4: What's your guess? 10
Too high! Try a lower number.

Attempt 5: What's your guess? 6
Too high! Try a lower number.

Attempt 6: What's your guess? 1

You got it! The correct number was 1.
You took 6 attempts.
Your score is: 340

Would you like to play again? (y/n): n
Thanks for playing! Goodbye!
```