

Software Unit Testing Report

Guess the Number game by using Test Driven Development (TDD) in Python Programming Language



August 25, 2023

Student Name: Amair Ahmed

STUDENT ID: S366847

CAMPUS: CASUARINA

Table of Contents:

[1. INTRODUCTION: 2](#_Toc143855401)

[2. Process: 3](#_Toc143855402)

[2.1 Functionality 1: (Guess number & How many attempts for guessing secret number) 3](#_Toc143855403)

[2.2 Functionality 2: (Provide hints to user) 5](#_Toc143855404)

[2.2 Functionality 3: (Display the success message and ask the user to play again or not) 6](#_Toc143855405)

[2.4 Functionality4: (Validate the input for a 4-digit number) 8](#_Toc143855406)

[2.5 Functionality 5: (Quit the Game) 9](#_Toc143855407)

[2.6 Functionality 6: (Test case for Invalid Length guess, Invalid short number, non-digit guess) 10](#_Toc143855408)

[3. Conclusion: 11](#_Toc143855409)

# INTRODUCTION:

The main objective of the task is to develop GUESS THE NUMBER game in python programming language by using automate unit test environment. participants try to figure out a secret 4-digit number using the supplied code. The game allows users to guess, and it offers tips to assist them restrict their options after each guess. Until the user correctly guesses the secret number or decides to give up, the game cycle continues. Users have the choice to continue playing or leave each game once it has ended. Furthermore, participants enter their own 4-digit predictions once the game starts with a randomly generated 4-digit secret number. The suggestions are then given based on how the predicted digits are positioned in relation to the digits of the secret number. An 'O' is shown if a guesses digit is in the right spot. An "x" is displayed if a predicted digit is accurate but in the wrong location. A '\_' is shown if the digit is absent from the secret number.

By entering "quit," users can exit the game at any moment and see their secret number and the amount of attempts they made. When the user correctly guesses the secret number, the script verifies that the input is correct, keeps track of how many times they have tried, and displays congratulatory messages. Until the user chooses not to play again, the game is played endlessly.

Demonstrating the phrase automated testing, which covers the automatic testing of the application, in which the developer will construct a unique and different script for the testing file using various automated testing tools. Following the creation of the testing file, the programmed file will automatically test the application and display the results. This is the basic principle of an automated testing tool.

In this project, I am using unit testing, which will automatically test the functioning of the game's key functionalities by inserting pass/fail situations into the test cases. Consider the game to be the unit of code. The core capability of unit testing is to test that unit of code using various test cases, analysing their behaviour and results.

We need to import the Unit test package module in testing file, which is already in the standard library, to do unit testing in Python. We don't need to install any other modules. Each of the test cases has been expanded further with appropriate screenshots.

# Process:

Guess number game has been developed using the Test-Driven Development (TDD) in python programming language by using the following steps and conditions.

I have written the different functions for the major functionality of the game in the guessing\_game.py file which is the major game coding file and I have written the test cases of these major functionalities into the other unit testing file called as test\_guessing\_game.py. test\_guessing\_game.py file will automatically test the behaviour and functionality of the game as I have written different test cases with different inputs and also provided different scenario.

# 2.1 Functionality 1: (Guess number & How many attempts for guessing secret number)

A user will attempt to guess a four-digit secret number. The computer provides you with tips to assist you in determining the number each time you guess. How many attempts you make are recorded by the game. You can type "quit" to end the game, and the computer will then reveal the secret number. The game notifies you of your victory and the number of attempts it required if you correctly predict the number. Overall, this game works by producing a secret number, asking your guesses, providing suggestions, and telling you of your results or desire to give up. Simultaneously, the testing file will also verify the functionality of the match input function by providing the different input options through the testing file and check the response of the function automatically through coding. Here are the screen shots of both game and testing file.

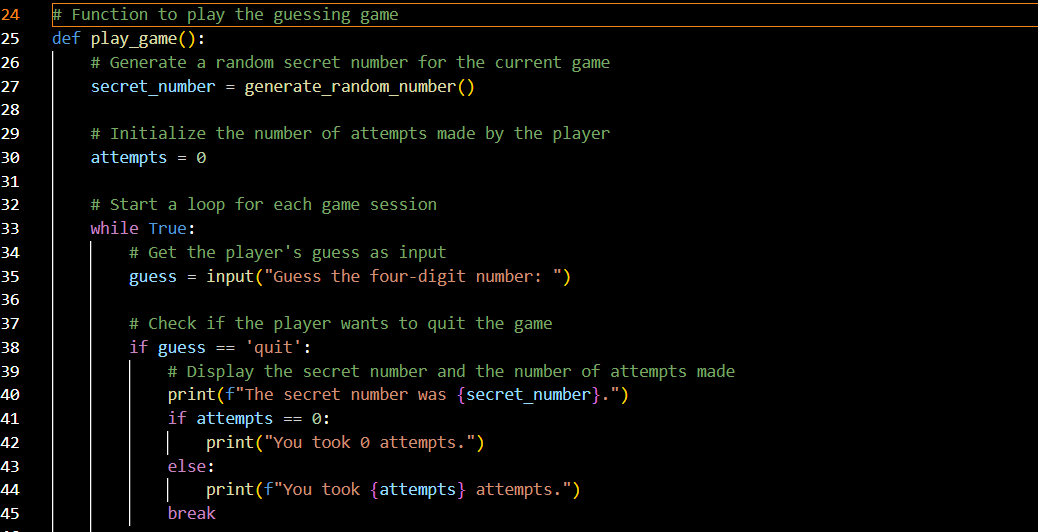


Figure 1: Guess number functionality from coding game

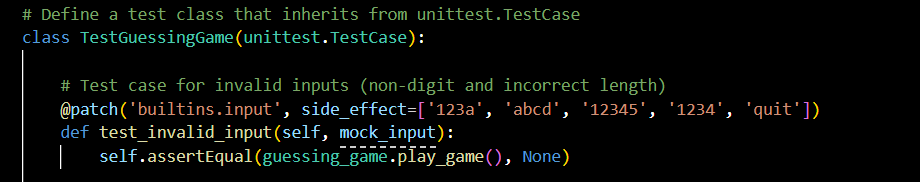


Figure 1.1: Guess number functionality from test\_guessing\_game.py file

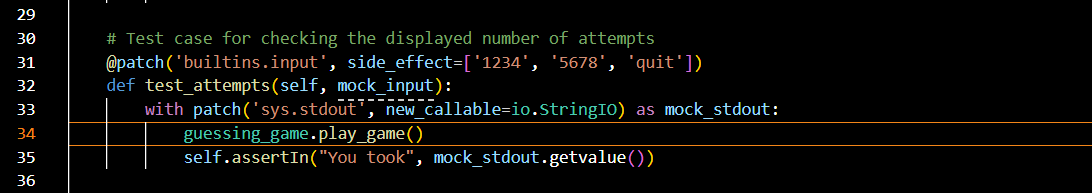


Figure 1.2: Attempts user took functionality from test\_guessing\_game.py file

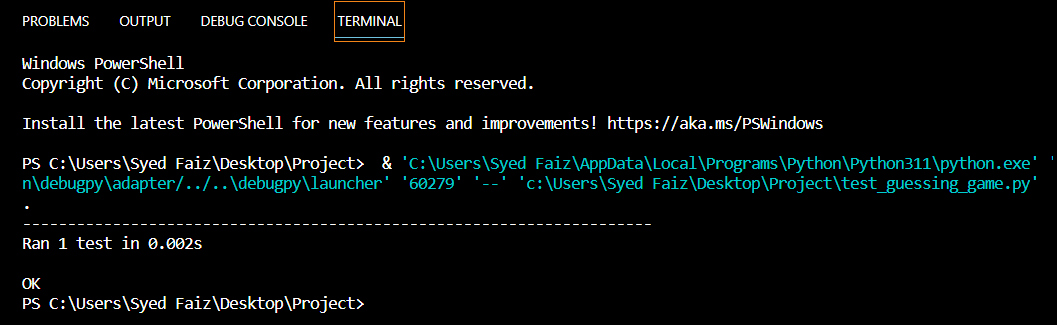


Figure 1.4: Test case passed for the functionality 1.

# 2.2 Functionality 2: (Provide hints to user)

After user written the guess number afterwards the game will provide the hints for the user to identify which position of the digit is correct. In which if the position is correct the game will give an indicator like ‘O’ that means the position of the digit is correct. On the other hand, side if the user enters the incorrect digit, then game will provide a indicator named as ‘X’ which means that the number is incorrect digit. Additionally, if the digit is absence, then it will give the user as ‘\_’ that means empty value. I have written the test case as well for this functionality that will clearly provide the hint to the user that user can change the value accordingly until the whole digit come correct. Here are the screen shots from both the files.

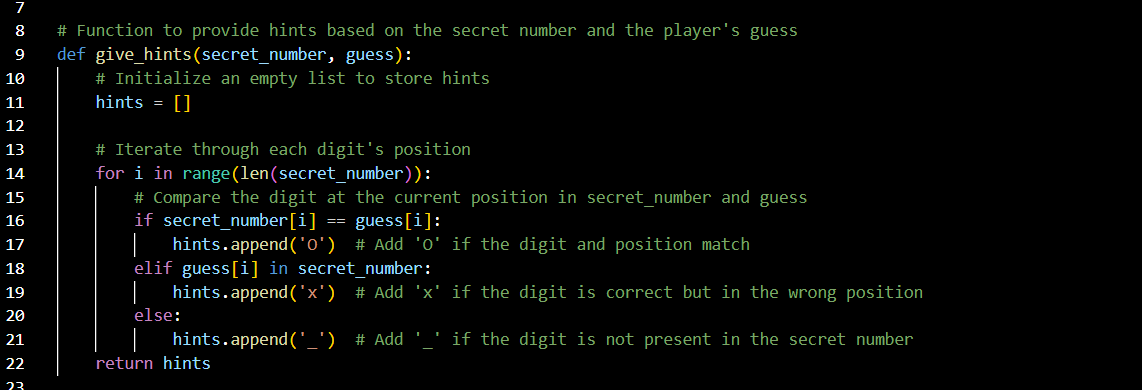


Figure 2: Game function of random input from the user

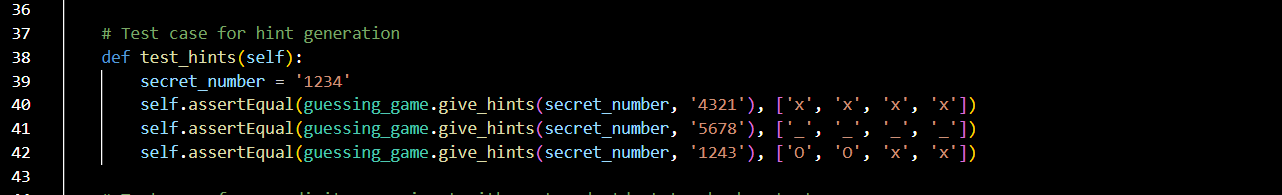


Figure 2.1: test case to verify the random input from the user

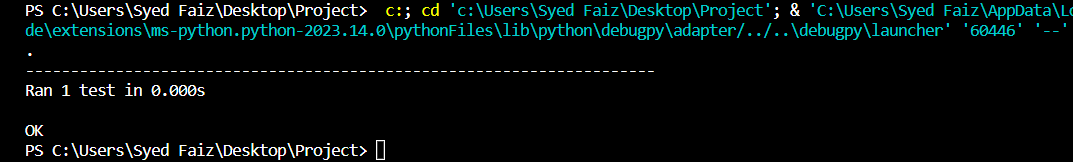


Figure 2.2: Output of the testing case

# 2.2 Functionality 3: (Display the success message and ask the user to play again or not)

After successful attempts of guessing the secret number, the game will provide a congratulatory message and game will ask the user to whether user wants to "Do you want to play again? (yes/no): “. If user write down ‘yes’ then the game will restart again with the same procedure asking for guessing the secret number and provide the hints accordingly to the user entered secret number after that it will shows up the congratulatory message and asked them to "Do you want to play again? (yes/no): " this will continuously repeated when the user says ‘no’ to play again. Moreover, after selecting ‘no’ the user will get the message such as ‘Thanks for playing the game’. Here are the screenshots from both the files.

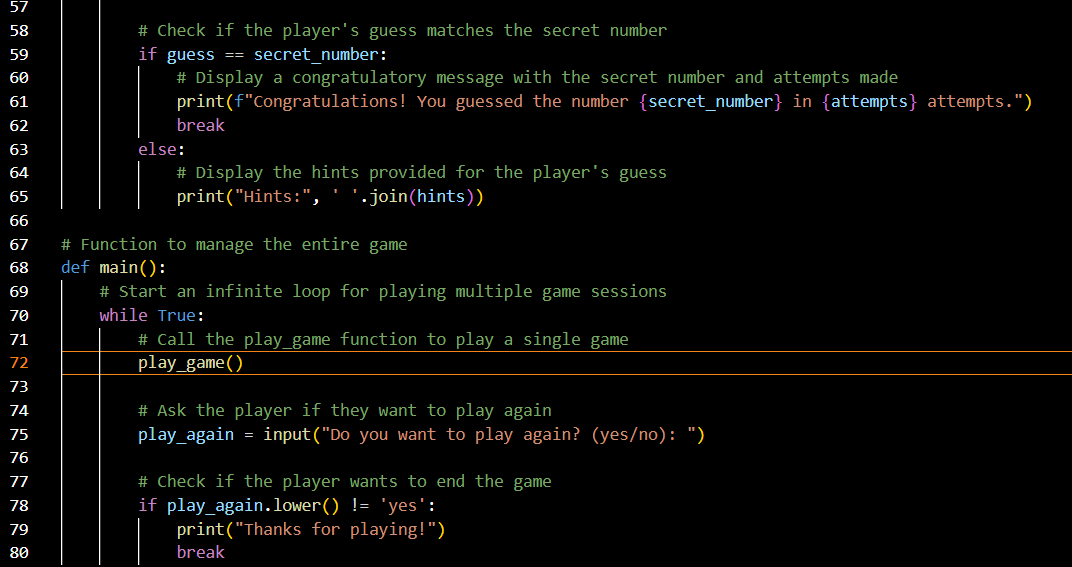


Figure 3: functionality for game play again or not

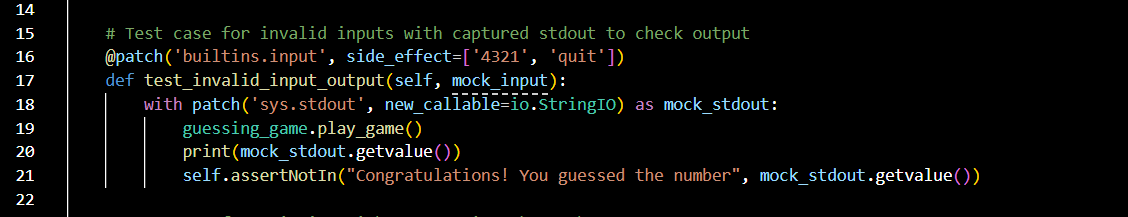


Figure 3.1 Test for the functionality from test\_guessing\_game.py file

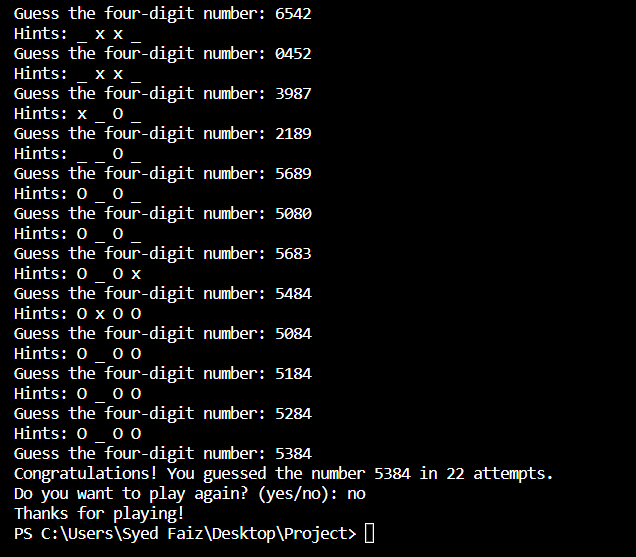


Figure 3.2 Output of the success message &play again functionality

# 2.4 Functionality4: (Validate the input for a 4-digit number)

While guessing the secret the user might enters the more digits instead of four digits then game will provide an error message to the user says that Invalid input. Please enter a four-digit number." Afterwards, the game will continue will there itself unless and until user doesn’t provide the 4 digits number. By having the this restricted the user will get know that they are doing mistake while playing the game. This will be beneficial for the user and improve the intangibility for the game simultaneously. Here the below screenshots from both files.

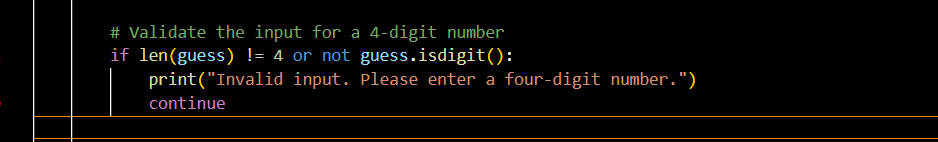


Figure 4: Invalid input from the user

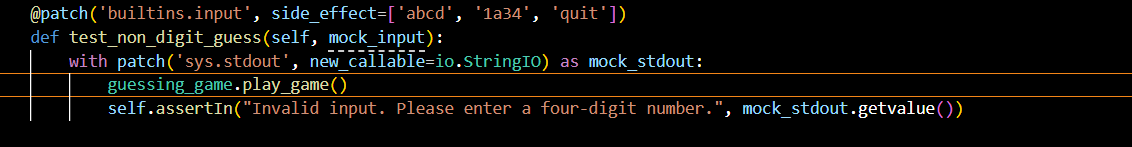


Figure 4.1: Test case for the Invalid Input

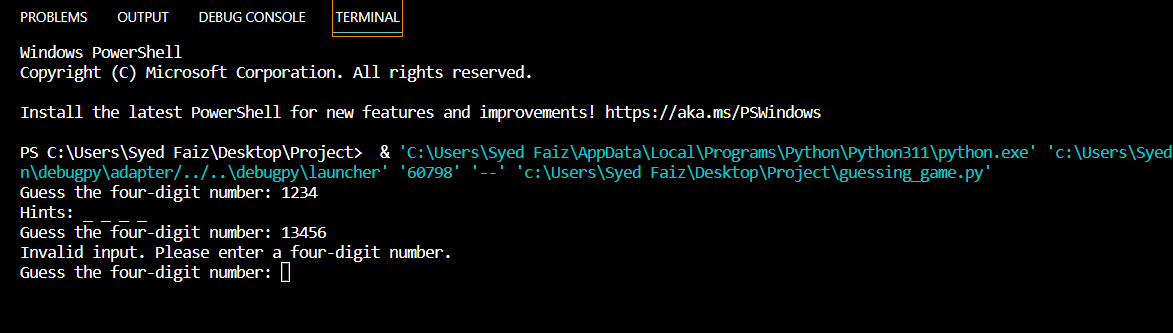


Figure 4.2: Output for the more than 4 digits entered by the user

# 2.5 Functionality 5: (Quit the Game)

This functionality is very useful for the user if the user doesn’t want to play the game and they no longer to interested to play they can quit the game anytime during the whole game. The user need s to just type ‘quit’ the game will directly quits and display the message says that ‘The secret number was ‘whatever the number is’ and it also says that how many attempts it will take ‘You took 1 attempt. Moreover, it will ask the user played again or not. If user say ‘yes’ then the game will continue as usually and if ‘no’ it will game ends.

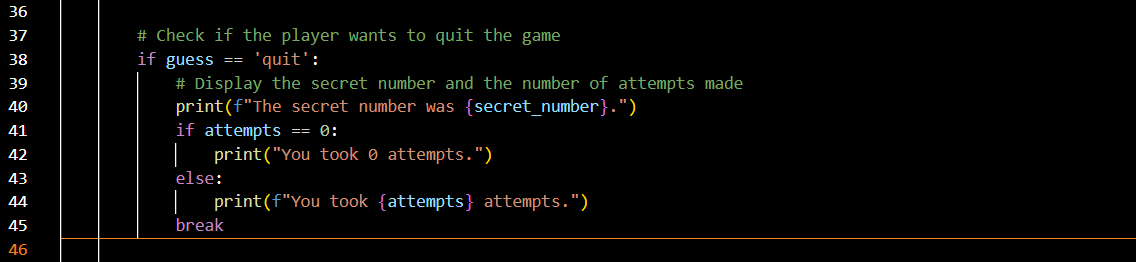


Figure 5: Functionality for quitting the game

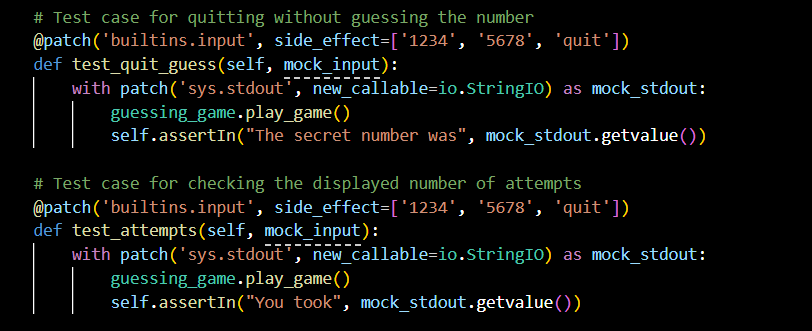


Figure 5.1: Test case for the quitting the game.

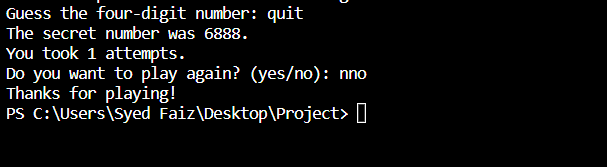


Figure 5.2: Output for quitting the game

# 2.6 Functionality 6: (Test case for Invalid Length guess, Invalid short number, non-digit guess)

This functionality is very efficient for the users who are making mistakes every time here and then. It will provide the number the error according the input such as if the user enters the invalid length guess input which means entering more digits to the standard 4 digits, then it will provide a message says that "Invalid input. Please enter a four-digit number." Similarly, if the user hits alphabetical letter as an input it will shows the same error and same for the short number guess.

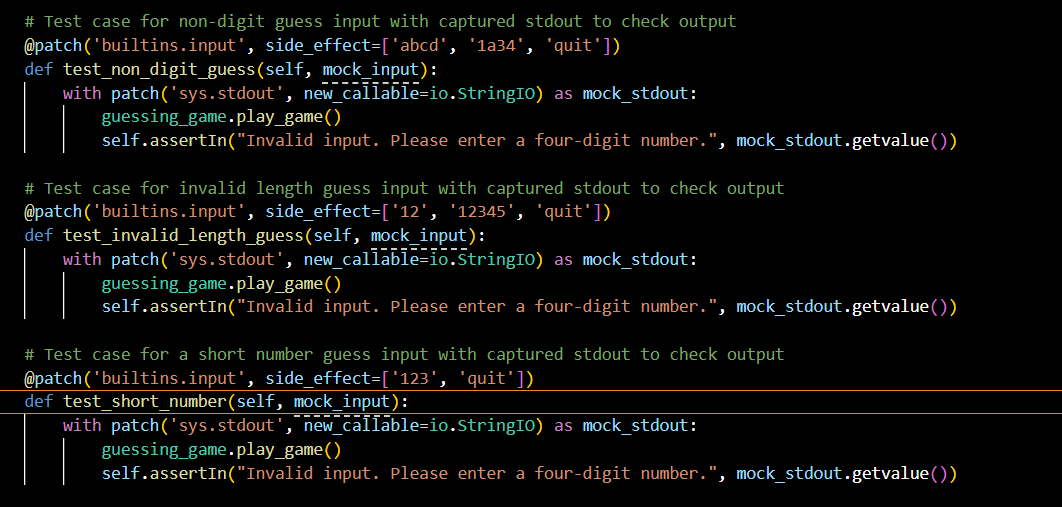


Figure 6: Test case for the above functionalities

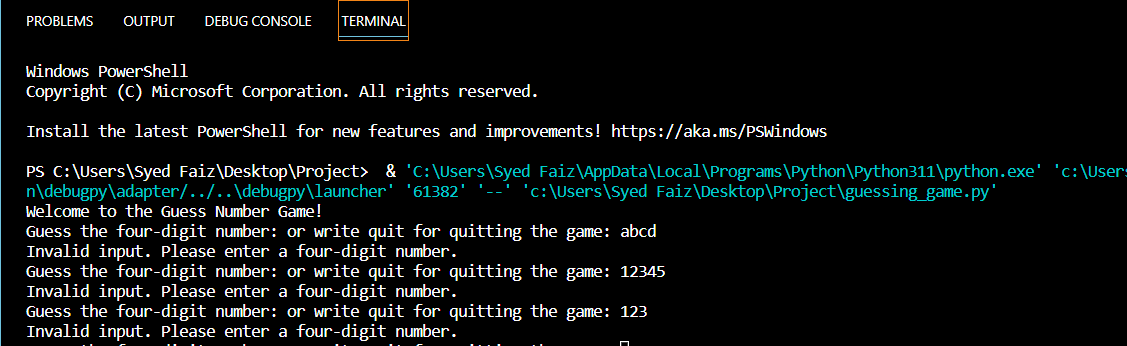


Figure 6.1: Output of a restrictions for invalid inputs from the user

# Conclusion:

Through this project, I successfully made a number guessing game that follows the rules we set. I used something called Test-Driven Development (TDD) and a tool called "unittest" to test the game automatically. This helped me make sure that the game works well and does what it should. TDD also helped me find and fix problems early. This project taught me the importance of testing and how it makes software strong and reliable.

This game also taught me about making things step by step using TDD. First, I tested my ideas, then I built the game. This way, I caught problems early and made a strong foundation for the game. As I worked on the game, I realized that TDD is difficult then I started doing one piece at a time then it made me feel more confident about each step I took. Even when there were challenges, like figuring out how to handle player guesses and give hints, TDD guided me in the right direction. It showed me that improving the game bit by bit and testing along the way is a smart way to build things. This project gave me a idea of how to write a code in efficient manner.

Please find the below GitHub link for my project:

<https://github.com/ahmedau024/Software-Unit-Testing>