INFORMATICS INSTITUTE OF TECHNOLOGY

GameInt

Doc 333 Programming Coursework

Assignment Cover Sheet

Course:

Foundation Certificate Programme

Unit Code and Description:

Introduction to Programming in Python – P1

Module leader:

Mr. Sudharshana Welihinda

Assignment Number: 01

Assignment Type: Individual coursework

Issue Date: 14th November 2022

Hand-in-Date: 12th December 2022

Deadline: on or before 9 AM

Name: P.R.S.Y.R.S.Jayasinghe

Student ID: 20221584

Acknowledgement

First I wish to express my sincere gratitude to our senior lecturer **Mr. Sudharshana Welihinda** and our tutorial lecturer **Ms. Salitha Dinushika** for guiding and instructing me to come up with my report. For me it was a unique experience to study, research and create a game through python coding.

Secondly I would like to thank my parents for giving me encouragement enthusiasm and invaluable assistance.

Without all this I might not be able to complete the report properly.

At last I apologize all other unnamed who helped me in various ways to have a good training.

I perceive as this opportunity as a big milestone in my career development. I strive to use gained knowledge and skills in the best possible way.

Table of Contents

List of tables

List of figures

1. Introduction to the problem

Coursework assignment is based on python programming language. The python code is to create a deduction game in which players alternately make guesses using a finite number of possibilities and use logic to determine which pegs their opponents have concealed.

1.1. Background

At one end of the gaming board, the code maker inserts four colored pegs covertly into the openings behind a screen. Here, the six colors will be represented by integer integers ranging from 1 to 6. Such as 1-White, 2-Blue, 3-Red, 4-Yellow, 5-Green, and 6-Purple, and will produce a four-digit number at random. This will stand in for the four hues that were chosen in secrecy.

The person who cracks the code makes several guesses by typing a 4-digit number after each one. This four-digit number will stand in for his guess's four colors and the position. The code maker uses smaller pegs to make guesses, and the system will place a '0' to represent a white peg to indicate a guess that is correct in terms of color but incorrect in terms of placement, and a '1' to indicate a black peg to indicate a guess that is correct in terms of both color and placement. If it is wrong color entirely then the system does not use any pegs and keep it as a ‘hash’ sign.

1.2. Objective

The main goal of this Python code is to design and develop a well-written, user-friendly deduction game.

2. Algorithm

01. Start

02. Initialize variables

03. Ask whether to start the game by the user

04. If user input ‘Yes’, ask for the name of the user

05. If user input ‘No’, display “Good bye…….See you again”, go to step 18

06. While number of Attempts<=8

07. Get a 4 digit code from user and validate

08. If Attempt = ‘0000’, display the error message and go to step 18

09. If Attempt = is>4 digits, display the error message and repeat from step 07

10. If Attempt is <1 or >6 display the error message and repeat from step 07

11. If Attempt is not an integer, display the error message and repeat from step 07

12. If step 09, 10 and 11 are true, check the validation

13. Display the validation

14. If validation is ‘1111’, display “You win!” else display “You lose!”

15. Ask user whether to restart (Yes or No)

16. If ‘Yes’ repeat from step 03

17. If ‘No’ display “Good bye”

18. End

3. Flowchart

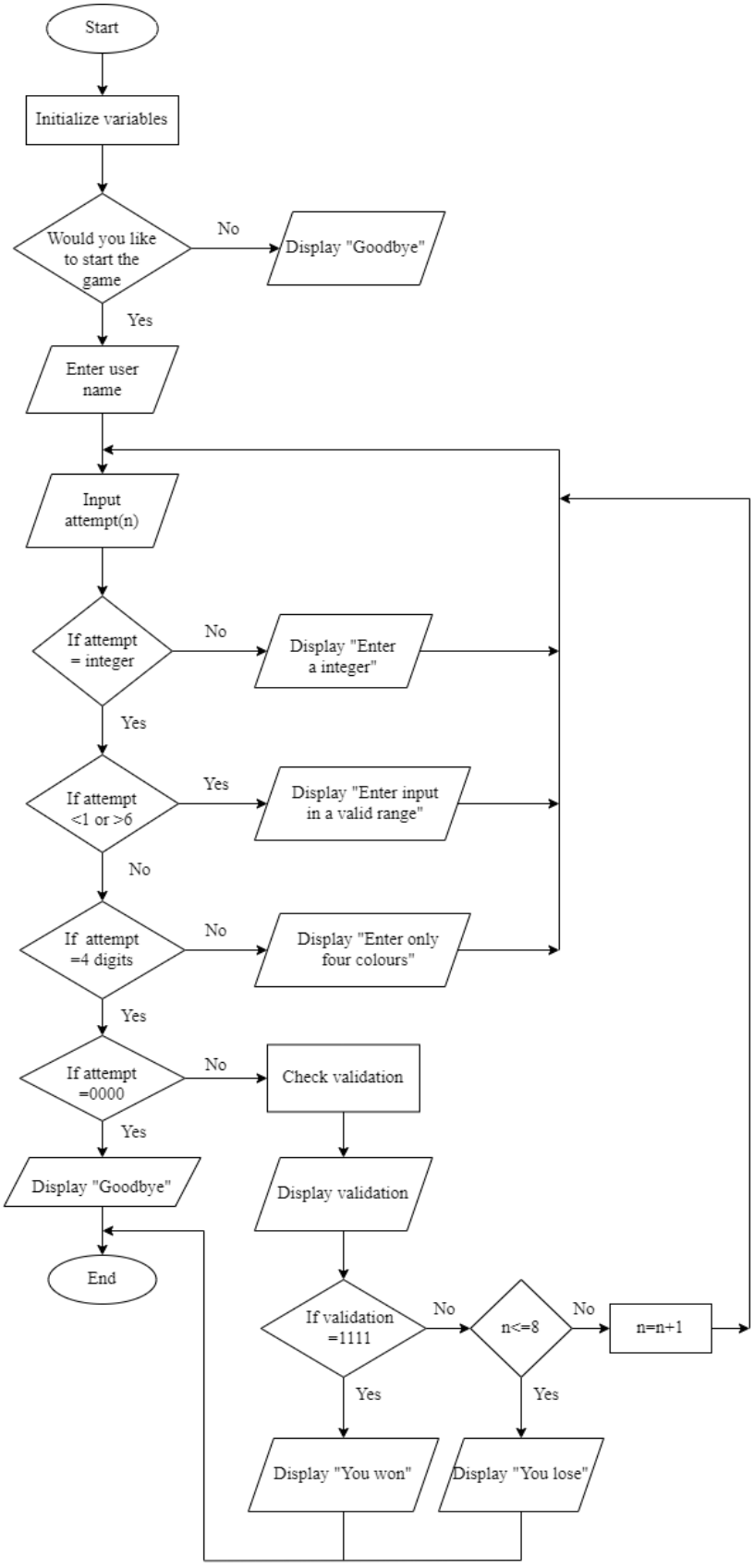

Figure 1:3 - Flowchart

4. Solution to the problem

With all the specifications, a game was written in Python. As an added feature, the wrong color peg in the wrong location is indicated with a hash sign after validation, and an additional try will be made if an invalid input is provided.

5. Table of test cases

Hidden Code	Test case	Input	Expected output	Actual output	Remarks
6333	01	1234	▪ Colours you guessed are: [1,2,3,4] ▪ Validation of guessed numbers are: [##1#]	▪ Colours you guessed are: [1,2,3,4] ▪ Validation of guessed numbers are: [##1#]	Pass
	02	12345	▪ Enter only 4 colours	▪ Enter only 4 colours	Pass
	03	ABcd	▪ Enter a valid integer	▪ Enter a valid integer	Pass
	04	6789	▪ Enter a value between range 1-6	▪ Enter a value between range 1-6	Pass
	05	0000	▪ Game ended Good Bye!!	▪ Game ended Good Bye!!	Pass

Table 1:5 - Test cases

6. Screenshots of different test cases

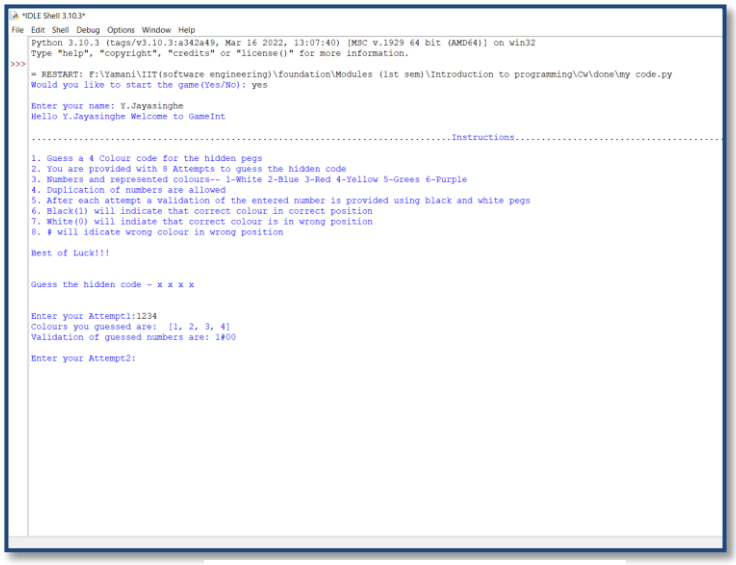

Figure 2:6 - Example for test case 01

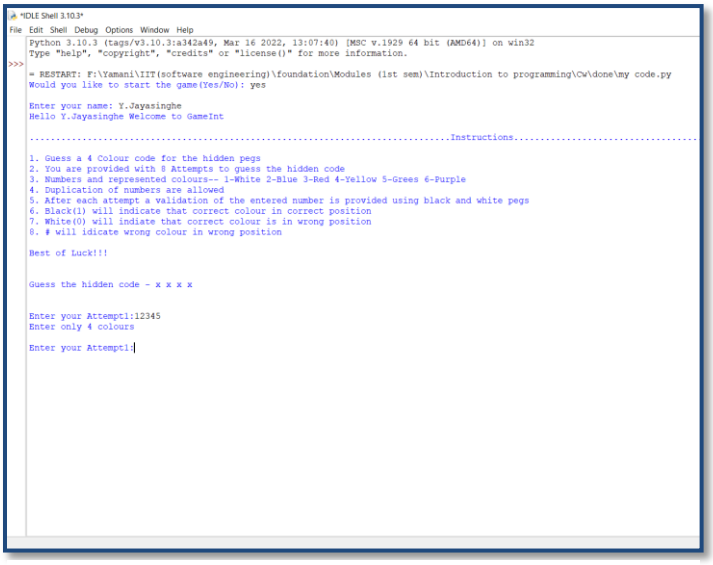

Figure 3:6 - Example for test case 02

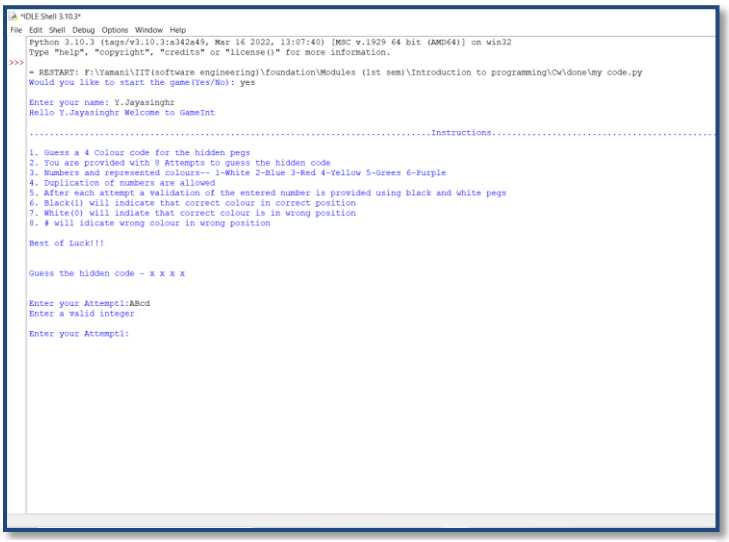

Figure 4:6 - Example for test case 03

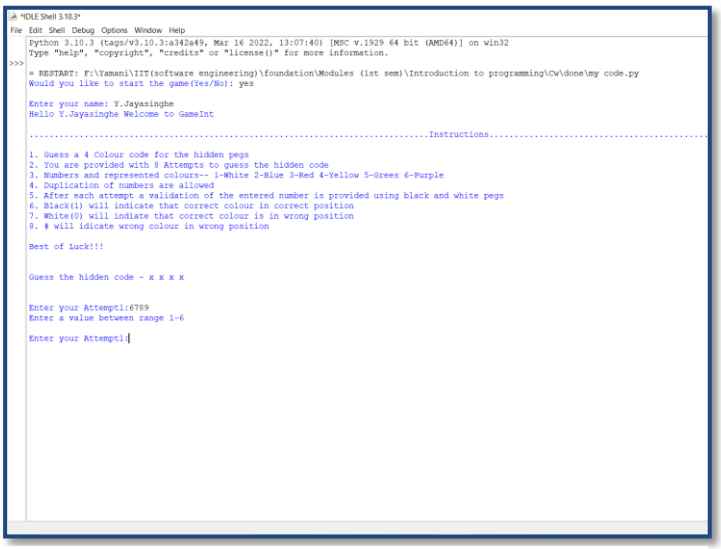

Figure 5:6 - Example for test case 04

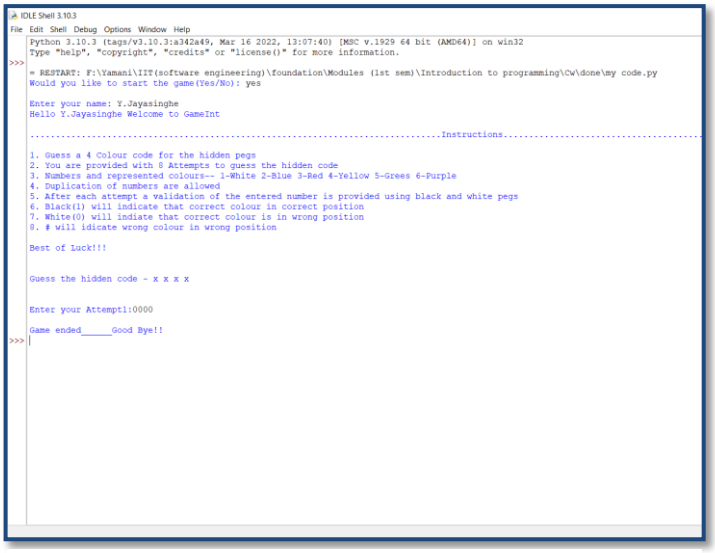

Figure 6:6 – Example for test case 05

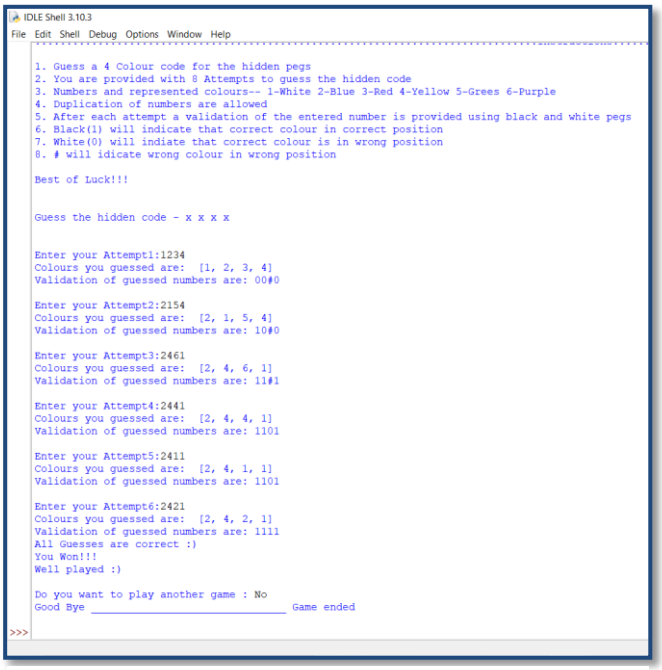

Figure 7:6 – Example for game not to be continued

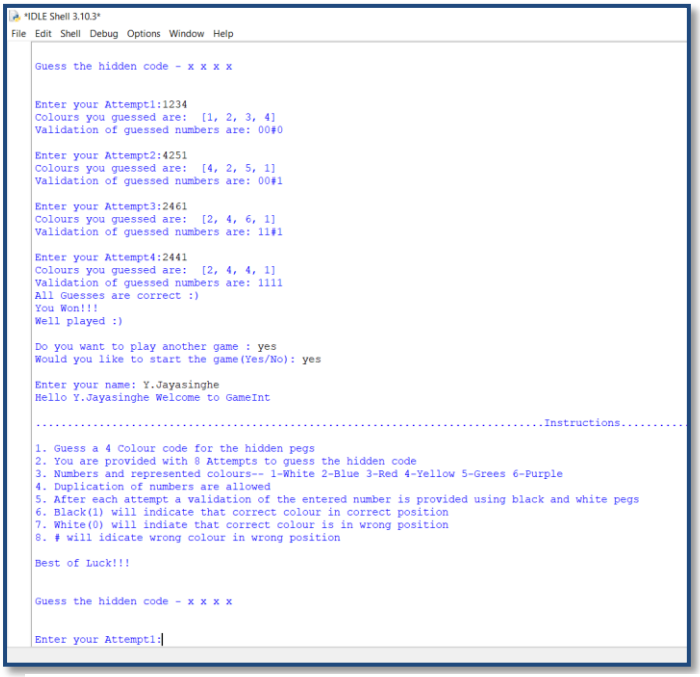

Figure 8:6 – Example for game to be continued

7. Conclusion

The fundamental mindset that this project seeks to cultivate is viewing the game as a component of a larger educational process. GameInt's coding was incredibly challenging, and there were numerous mistakes. Before a final, effective answer could be identified, multiple systems had to be written in various methods. The indentation and dedentation had problems as well.

Because of these factors, it is advised that anyone wishing to reproduce this game begin by creating the code simple. It is recommended that it be simpler to get specific features to function. When debugging the code, using functions made it simpler to spot where issues were happening. It maintained the code cleaner and more structured.