INFORMATICS INSTITUTE OF TECHNOLOGY

# Foundation Certificate in Higher Education

# DOC 334: Introduction to Programming 2

# Coursework

| | |
|---|---|
| **Module** | DOC 334 Introduction to programming 2 |
| **Module Leader** | Mr. Sudarshana Welihinda |
| **Assessment Type** | Individual Coursework (ICW) |

| | |
|---|---|
| **Student Id** | 20210010 |
| **Student Name** | Sarith Wijesundera |

# 1. Abstract

This report is a part of our second semester's coursework for the "DOC 334: Introduction to Programming 2" module. The main task of the coursework is to create a console python program to implement the game "Hangman". Therefore, the purpose of this report is to give a description of the coursework problem and to show the solution that I have implemented for the coursework. This report includes all the codes as a text, test cases, screenshots of the test cases / outputs, and every other requirement that has been asked to indicate in the report. In addition, this report was done according to the good practices of report writing covered by "DOC 311: Academic Skills for Higher Education" module and the good word processing skills learned from "DOC 314: Introduction to Information Technology" module.

# 2. Acknowledgements

First and foremost, I would like to take this opportunity to be grateful to the management of IIT and staff for successfully arranging the foundation program and creating a suitable environment for us to lift our both academic and general skills by the guidance of our reputable lecturers.

I would like to express my gratitude to our Course Leader of the foundation program, Ms. Shyani Siriwardene, and our Level Coordinator, Mr. Namal Malalasena for their dedication to ensuring the best experience for the students even through this hard time.

I am really thankful and fortunate enough to have Mr. Sudarshana Welihinda as the Module Leader and Mr. Nishan Saliya as the lecturer for the module "DOC 334: Introduction to Programming 2". The knowledge and the honourable guidance they gave us through this time have helped me to complete the coursework successfully. In addition, I'd like to thank them again and again for sparing their time to clarify our doubts patiently whenever we'd ask them.

I would also like to pay my tribute to Ms. Tharushi Amarasinghe and Ms. Keerthiga Rajenthiram, for helping us and sharpening our skills throughout the Tutorial and the self-study sessions of the "DOC 334: Introduction to Programming 2" module and providing us more and more information regarding the subject by giving constant encouragement, support, and guidance.

Last but not least, I would like to thank our fellow batch-mates for helping us through the coursework. At the same time, I would like to thank all my family members for assisting me through this tough time and cooperating with me to conduct the coursework without any interruption or any disturbance even though I have done this inside my house.

# 3. Table of Contents

# 4. List of Tables

# 5. List of Figures

# 6. Body of The Report

## 6.1. Description of The Problem Statement

It is required to develop a console Python 3.x program to implement the game “Hangman”. Hangman is a single player game, where a player gets some empty spaces to guess a word within a limited number of turns. The turns are equal to the length of the word. If the player enters a letter which is in the word, then the number of turns will not be change and that specific letter in the word will only be displayed along with other empty spaces. Or else, if the player input letter is not in the word, then the number of turns will reduce by one and nothing will be changed in the empty spaces. So, simply if the player could guess the correct word within the given number of turns, then the player will win. Or else, if the player runs out of the turns before guessing the correct word, then the player will lose.

Moreover, there should be 20 stored words in the program, which will appear randomly during the game. And, each word should have different number of turns based on the length of the word. In addition, they have asked to display a small hint for the player to guess the word. At the same time, it is required to collect the player information such as player’s name, word guessed, turns provided, turns used and win/lost status.

Our task in this coursework is to create the above game. They have given us the freedom to decide the home menu of the game and the interface of the console, based on the above requirements. And, if we have any assumptions, we must clearly state them too. Furthermore, it is required to create a file base or a database to view the past game play history of the players. If we are using a database, then the username must be “root” with no passwords and the server must be “localhost”. At the same time, we must provide means to import/restore our current database. If we are using a text file to store the past records, then the extension must be .TXT. Since this is a single player game, the player must be able to play the game as many times he/she likes. But also, the player must be able to exit from the game anytime they wish. Moreover, the program should keep a record of wins and losses of each game play. At the end of the game play, it should display the game stats such as total number of games played, total wins and total loses. As a challenge activity, we can display the game stats in a HTML file.

## 6.2. Description of The Solution I have developed

As the solution for the above problem statement, I have developed a console Python 3.x program which fulfills the above requirements. To start things off, if we get the menu system, it begins by welcoming the player to the game. Then, to collect the player information, the program is asking user to enter the player's name. In addition, to make the "Hangman" game more interesting, I have developed two levels. They are level 'easy' and level 'hard'. The difference between these two levels is, the hints in the level 'easy' are not too difficult to guess and the hints in the level 'hard' are a bit difficult to guess. Therefore, I have clearly stated that assumption in the menu system and then asking the user to enter the preferred level he/she wants. After that, the program is asking from the user to enter the number of rounds he/she wants to play because it is clearly mentioned in the problem statement that the user must be able to play the game as many times, he/she likes. Next, the program is displaying three main instructions / game tips for the user to continue with the game. Those instructions include the followings. Since, it is mentioned as a requirement in the problem statement, the player can leave from the game anytime he/she wish by typing the word "exit" after the game has started. As an addition, I have also used the word "skip" as a command to skip a round because, when the words randomly appear during the game, the same word can be appeared more than once. Therefore, in such situations, the user can simply enter the word "skip" and go to the next round. After the instructions, the program will randomly generate and display a record number for the player which is in between 0 and 9999. The menu interface will end with it.

To start the game, I have used three main functions. The first function includes the whole game process and the second function includes the process of the database. The first function will be called in a loop from the second round onwards. Also, as an addition, when moving from one round to another, the program will ask the user's consent. If it is a 'yes' then the user can move to the next round, and if it is a 'no' then the game will end. At the same time, to validate the program, I have used some error validation statements. Therefore, based on the situation, if the user enters some invalid input, then it will display an error message with the reason for the error and then the program will end. After the game play ends, the game statistics such as total rounds played, total rounds skipped, total wins and total loses will display. At the same time, I

have used the third function to display the game statistics in a HTML file which has mentioned as a challenge activity in the problem statement.

When it comes to the database function, I have developed it to restore per session. As the database, I have used ‘db_test_python’ and according to the problem statement, the host is ‘localhost’, the username is ‘root’ and it has no passwords. This function stores the game records in a data table called “game_history”. The “game_history” data table includes the columns ‘Rec_No’, ‘Player_Name’, ‘Total_Rounds’, ‘Total_Skips’, Total_Wins’, and ‘Total_Loses’ where ‘Rec_No’ acts as the primary key. At the beginning, this function checks whether there’s a database called ‘db_test_python’. If not, it will create a new database with that name. In that case, this database can be restored by any computer.

## 6.3. Program Codes

```
#Importing packages
import random
import mysql.connector
import webbrowser

#Create functions
#Function 1
def game(word_list,lists,level,store,name):
    "This function gets a random word from the list and gives a hint to the pl
ayer to guess the word based on the difficulty level"
    random.shuffle(word_list)
    word = random.choice(word_list)

    #Defining global variables
    global wins
    global loses
    global skip

    #The hints for the level 'easy'
    if level == "easy":
        if word == "banana":
            hint = "This is a yellow fruit"

        elif word == "hen":
            hint = "They lay eggs"

        elif word == "apple":
            hint = "if you eat this once a day, you can keep your doctor away"

        elif word == "cat":
            hint = "Are you bothered by rats? Then you definietly need this"

        elif word == "telephone":
            hint = "You can't survive a day without using this"

        elif word == "laptop":
            hint = "Type of a computer which you can take anywhere with you"

        elif word == "bottle":
            hint = "This allows you to carry anything liquid"

        elif word == "technology":
            hint = "This is the main reason which connects the world together"

        elif word == "tiktok":
            hint = "This is a social media app where you can upload short vide
os"
```

```
        elif word == "elephant":
            hint = "The largest animal on the land"

        elif word == "school":
            hint = "We get our primary education from here"

        elif word == "butterfly":
            hint = "You can find this living creature commonly in a garden ful
l of flowers"

        elif word == "internet":
            hint = "You need this to connect with the world"

        elif word == "russia":
            hint = "The largest country in the world"

        elif word == "wardrobe":
            hint = "You keep your clothes in here"

        elif word == "mars":
            hint = "The only place in the universe which has water other than
the Earth"

        elif word == "antartica":
            hint = "highest, driest, coldest and windiest continent on Earth"

        elif word == "alarm":
            hint = "We use this to wake up early"

        elif word == "aeroplane":
            hint = "By using this, we can travel anywhere on the world"

        elif word == "thieves":
            hint = "We can't sleep peacefully because of them"

    #The hints for the level 'hard'
    elif level == "hard":
        if word == "banana":
            hint = "This believed to be as the world's first fruit"

        elif word == "hen":
            hint = "We raise them on farms"

        elif word == "apple":
            hint = "This fruit contains 25% of air"

        elif word == "cat":
            hint = "The only mammal who doesn't taste sweetness"

        elif word == "telephone":
            hint = "This was invented by Alexander Graham Bell"
```

```
        elif word == "laptop":
            hint = "These can be used in a smaller space than an ordinary desk
 computer"

        elif word == "bottle":
            hint = "This comes as plastic, glass, and metal"

        elif word == "technology":
            hint = "People use this to make their daily lives easier"

        elif word == "tiktok":
            hint = "This app is the most downloaded app on the Apple app store
 in 2019"

        elif word == "elephant":
            hint = "This animal won't forget anything"

        elif word == "school":
            hint = "If a country gets more of these, literacy will increase"

        elif word == "butterfly":
            hint = "This insect use their feet to taste"

        elif word == "internet":
            hint == "More than half of the world's population is using this se
rvice"

        elif word == "russia":
            hint = "This country has the biggest nuclear arsenal in the world"

        elif word == "wardrobe":
            hint = "You open this before going to a party"

        elif word == "mars":
            hint = "This is known as the 'Red Planet'"

        elif word == "antartica":
            hint = "This is the largest ice store in the world"

        elif word == "alarm":
            hint = "You need it everyday but you hate it every morning"

        elif word == "aeroplane":
            hint = "An invention of two brothers, which made an evolution in t
he world"

        elif word == "thieves":
            hint = "Them are the reason why we put cctv cameras"


    print("Word   : ",end='')
    for line in range(len(word)):
```

```
        print("_",end=' ')

    print("\n",len(word),"turns remain")
    print("**Hint :",hint)
    count = len(word)
    for c in range(20):
        letter = input("\nLetter : ")
        if letter == "exit":
            print("\nThanks for playing with us,",name,"! \nSee you Again!")
            print("\n",special*79)

            #Game Stats
            total = wins + loses + skip
            print("\nGame Statistics of",name,"\n")
            print("\tTotal Rounds Played  :",total)
            print("\tTotal Rounds Skipped :",skip)
            print("\tTotal Wins  :",wins)
            print("\tTotal Loses :",loses)
            print("\n**NOTE : The round you exit, will not be count to the tot
al rounds")
            create_db()
            html_file()
            exit()

        elif letter == "skip":
            skip += 1
            print("\nYou skipped the round")
            print("The Word is",word,"\n")
            break

        elif letter.isnumeric() == True:
            print("\nInvalid Input! \nLetter cannot be a number! \nPlease Try
Again!")
            create_db()
            exit()

        if letter in lists:
            count -= 1
            print("Word   :",end=' ')
            for i in range(len(word)):
                if word[i] in lists:
                    print(word[i],end=' ')

                elif word[i] not in lists:
                    print("_",end=' ')

            if letter in word:
                count += 1
                print("\n",count,"turns remain")
                continue

            elif letter not in word:
```

```
                print("\n",count,"turns remain")
                if count == 0:
                    loses += 1
                    print("\noops....You Lost!")
                    print("The Word is",word,"\n")
                    print("Better luck next time!\n")
                    break
                continue


        lists.append(letter)
        count -= 1
        print("Word   :",end=' ')
        for i in range(len(word)):
            if word[i] in lists:
                print(word[i],end=' ')

            elif word[i] not in lists:
                print("_",end=' ')


        if letter in word:
            store.append(True)
            count += 1
            print("\n",count,"turns remain")

        elif letter not in word:
            print("\n",count,"turns remain")


        if count == 0:
            loses += 1
            print("\noops....You Lost!")
            print("The Word is",word,"\n")
            print("Better luck next time!\n")
            break

        if store.count(True) == len(set(word)):
            wins += 1
            print("\nCongratulations....You Won!")
            print("The Word is",word,"\n")
            break


#Function 2
def create_db():
    "This function creates a table in the database and inserts data into it"
    "The data will include the record number, Player name, Total rounds played
, Total rounds skipped, Total wins and Total loses"
    "These data will be stored per session"

    #Defining global variables
```

```
    global wins
    global loses
    global skip
    global total
    total = wins + loses + skip


    db = mysql.connector.connect(
        host = "localhost",
        user = "root",
        password = ""
        )

    cursor = db.cursor()

    cursor = db.cursor()

    cursor.execute("SHOW DATABASES")

    #Storing the available databases in a list
    for databases in cursor:
        database_list.append(databases)

    #Creating a database, if that database is not in the database_list
    if ("db_test_python",) not in database_list:
        cursor.execute("CREATE DATABASE db_test_python")


    conDict = {'host':'localhost',
               'database':'db_test_python',
               'user':'root',
               'password':''}

    db = mysql.connector.connect(**conDict)

    cursor = db.cursor()

    cursor.execute("SHOW TABLES")

    #Storing the available data tables in a list
    for table_name in cursor:
        table_list.append(table_name)

    #Creating a data table, if that data table is not in the table_list
    if ("game_history",) not in table_list:
        cursor.execute("CREATE TABLE Game_history (Rec_No INT AUTO_INCREMENT P
RIMARY KEY, Player_Name VARCHAR(200), Total_Rounds INT(200), Total_Skips INT(2
00), Total_Wins INT(200), Total_Loses INT(200))")

    #Inserting data into the database
    mySQLText = "INSERT INTO game_history (Rec_No, Player_Name, Total_Rounds,
Total_Skips, Total_Wins, Total_Loses) VALUES (%s,%s,%s,%s,%s,%s)"
```

```
    myValues = (rec_no, name, total, skip, wins, loses)
    cursor.execute(mySQLText, myValues)

    db.commit()
    print("\n",cursor.rowcount,"Record Added")

    db.close()


#Function 3
def html_file():
    f = open("game_stats.html","w")

    template = f"""<html>
    <head>
    <title>Game Play History</title>
    </head>
    <body>
    <h2>Game Statistics of {name}</h2>

    <p>Total Rounds Played : {total}</p>
    <p>Total Rounds Skipped : {skip}</p>
    <p>Total Wins : {wins}</p>
    <p>Total Loses : {loses}</p>

    </body>
    </html>"""

    f.write(template)
    f.close()

    webbrowser.open("game_stats.html")

#----------------Main Program----------------
#Create Variables
word_list = []
letter = 0
lists = []
count = 0
store = []
rounds = 0
choice = 0
hint = 0
level = 0
name = 0
game_name = 0
special = "-"
f = 0
template = 0
wins = 0
loses = 0
skip = 0
```

```
total = 0
rec_no = 0
table_list = []
database_list = []

#Storing words
word_list = ["banana","hen","apple","cat","telephone","laptop","bottle","techn
ology","tiktok","elephant","school","butterfly","internet","russia","wardrobe"
,"mars","antartica","alarm","aeroplane","thieves"]

#Menu
game_name = "Welcome to the game HANGMAN!"
print("\n",game_name.center(79,"~"),"\n")
print("We have 20 words for you to guess --
- Grab a coffee and be ready to explore your knowledge!\n\n")

#Asking for the player name
name = input("Hello player! What's your name? : ")
print("\n")

#Asking for the difficulty level
print("Game Levels, \n \teasy --> The hints are EASY to guess \n \thard --
> The hints are HARD to guess\n")
level = input("Which level do you prefer? (easy/hard)  : ")
print("\n")

#Checking the validity
if level == "easy" or level == "hard":
    pass
else:
    print("Invalid Level! \nLevel must be either 'easy' or 'hard' only \nPleas
e Try Again!")
    create_db()
    exit()

#Asking for the number of rounds
rounds = input("How many rounds of words do you want to play? : ")

#Checking the validity
if rounds <= "0" or rounds.isnumeric() == False:
    print("\nInvalid Number of Rounds! \nThe number of rounds must only be gre
ater than '0' \nPlease Try Again!")
    create_db()
    exit()
else:
    pass

#Displaying the instructions
print("\n**NOTE : Type only one letter at a time")
print("**NOTE : You can exit from the game anytime you wish by typing the word
 'exit'")
```

```
print("**NOTE : If you want to skip a round, then simply type the word 'skip'"
)

#Displaying the record number of the player
rec_no = random.randrange(0,9999)
print("\nPlayer Record Number : ",rec_no)
print("\n",special*79)

#Calling the function for the 1st round
print("\nThis is round 1\n")
game(word_list,lists,level,store,name)

#Calling the function for the remaining rounds
for a in range(1,int(rounds)):
    choice = input("Do you wish to go to the next round? (yes/no) : ")
    if choice == "yes":
        print("\n",special*79)
        print("\nThis is round",a+1,"\n")
        letter = 0
        lists = []
        count = 0
        store = []
        game(word_list,lists,level,store,name)

    elif choice == "no":
        break

    else:
        print("\nInvalid Input! \nPlease try again by enter either 'yes' or '
no' only!\n")
        create_db()
        exit()

#Good-Bye message
print("\nThanks for playing with us,",name,"! \nSee you Again!\n")
print("\n",special*79)


#Displaying Game Stats
total = wins + loses + skip
print("\nGame Statistics of",name,"\n")
print("\tTotal Rounds Played  :",total)
print("\tTotal Rounds Skipped :",skip)
print("\tTotal Wins  :",wins)
print("\tTotal Loses :",loses)

#Calling the database function
create_db()

#Opening the HTML file which contains the game stats
html_file()
```

## 6.4. Test Cases

*Table 1 - Test Cases*

| Test Case # | Scenario | Input | Expected Output | Actual Output | Remarks |
|---|---|---|---|---|---|
| 1 | Entering the player's name | Adding any character as the name | Going to the next part (level) | Went to the next part (level) | Pass |
| 2 | Entering the preferred level | Inserting "hard" | Executing to the next part (No of rounds) | Executed to the next part (No of rounds) | Pass |
| 3 | Entering the preferred level | Inserting "easy" | Executing to the next part (No of rounds) | Executed to the next part (No of rounds) | Pass |
| 4 | Entering the preferred level | Inserting any character other than "easy" and "hard" | Displaying an error message | An error message displayed | Pass |
| 5 | Entering the number of rounds player wants to play | Inserting any number which is greater than zero | Displaying the instructions, player rec_no and starting the first round | The instructions and the player rec_no displayed and the first round started | Pass |
| 6 | Entering the number of rounds player wants to play | Inserting any number which is equal or less than zero | Displaying an error message | An error message displayed | Pass |
| 7 | Entering a correct letter for the guess word | Inserting any letter which is in the word | The inserted letter will be revealed and the no of turns won't be changed | The inserted letter was revealed and the no of turns didn't change | Pass |
| 8 | Entering a wrong letter for the guess word | Inserting any letter which is not in the word | There won't be any change in the empty spaces and the no of turns will be reduced by one | No change has happened to the empty spaces and the no of turns was reduced by one | Pass |
| 9 | Entering a number for the guess word | Inserting a number | Displaying an error message | An error message displayed | Pass |
| 10 | Guessing the correct word | Inserting the letters which are in the word | Displaying a "won" message | A "won" message displayed | Pass |
| 11 | Guessing a wrong word | Inserting the letters which are not in the word | Displaying a "lost" message | A "lost" message displayed | Pass |
| 12 | Player desire to go to the next round | Inserting "yes" | The next round will be started | The next round started | Pass |
| 13 | Player desire to go to the next round | Inserting "no" | Thank you message and the game statistics will be displayed in the console as well as in a HTML file and the game stats will be added to the database | Thank you message and the game statistics were displayed in the console as well as in a HTML file and the game stats were added to the database | Pass |

| | | | | | |
|---|---|---|---|---|---|
| 14 | Player desire to go to the next round | Inserting any word except “yes” and “no” | Displaying an error message and the game stats will be added to the database | An error message displayed and the game stats were added to the database | Pass |
| 15 | Skipping a round | Inserting “skip” | Player’s desire to go to the next round will be asked and it will continue based on the user input | Player’s desire to go to the next round was asked and continued | Pass |
| 16 | Exit from the game | Inserting “exit” | Thank you message and the game statistics will be displayed in the console as well as in a HTML file and the game stats will be added to the database | Thank you message and the game statistics were displayed in the console as well as in a HTML file and the game stats were added to the database | Pass |

## 6.5. Screenshots of Test Case Results / Outputs

**Test Case #1 -** *Adding any character as the name*

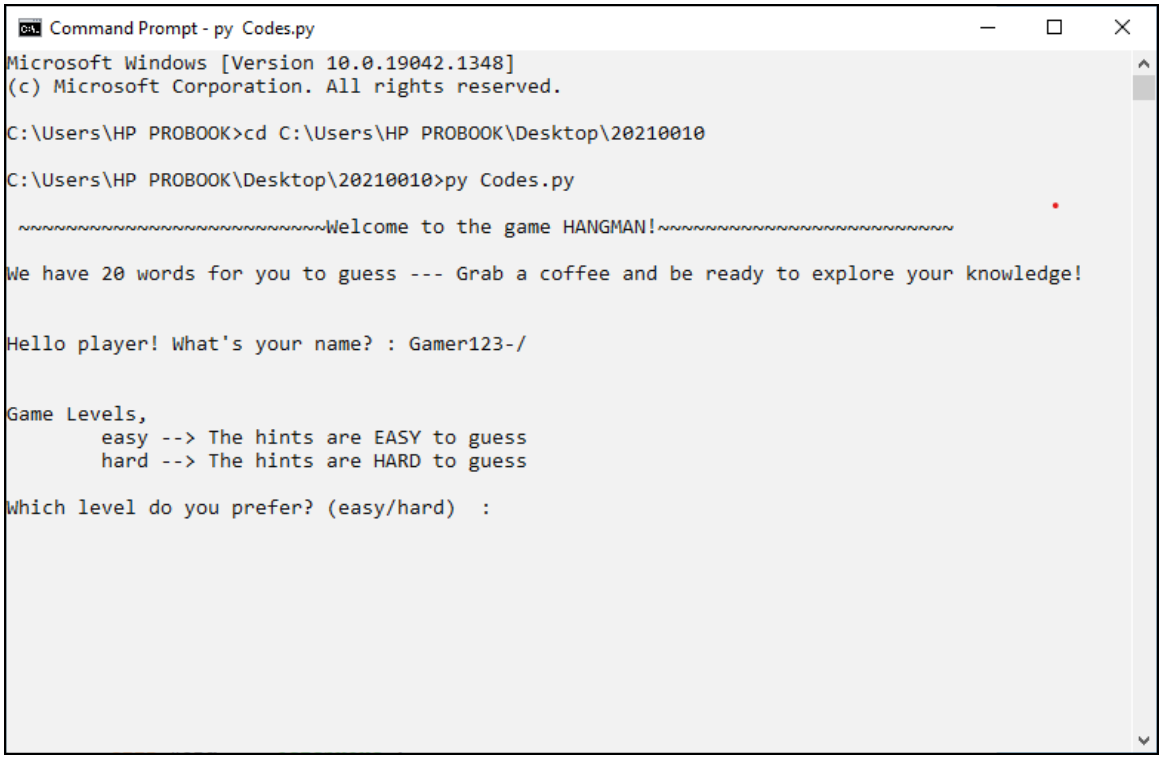


*Figure 1 – Test Case#1 Adding any character as the name*

**Test Case #2** – *Inserting "hard"*

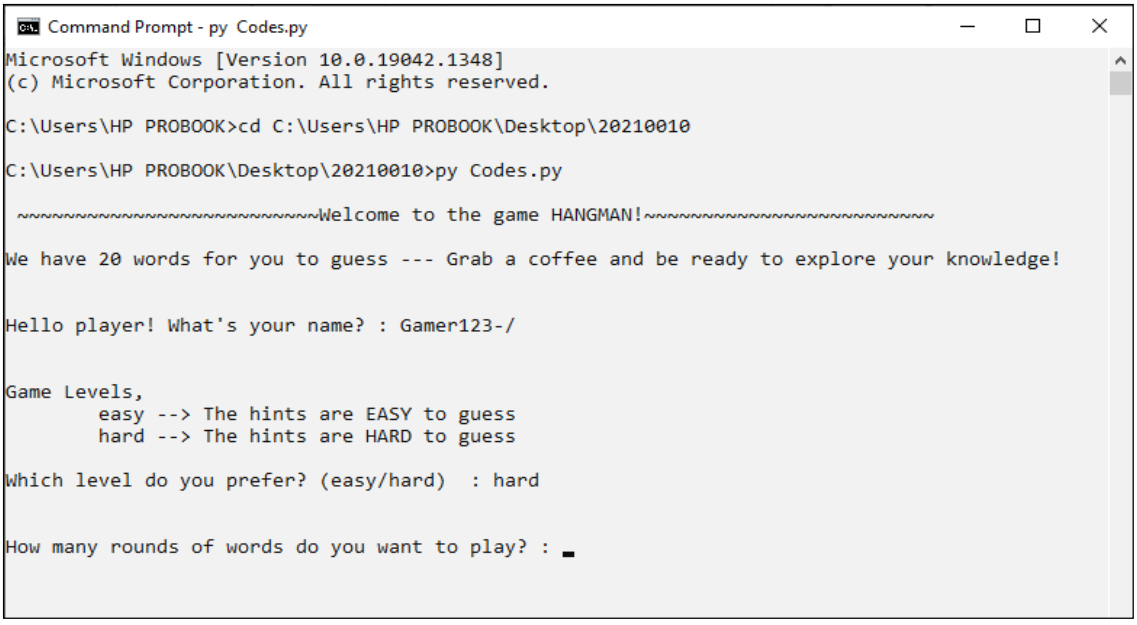


*Figure 2 – Test Case#2 Inserting "hard"*

## **Test Case #3** – *Inserting "easy"*

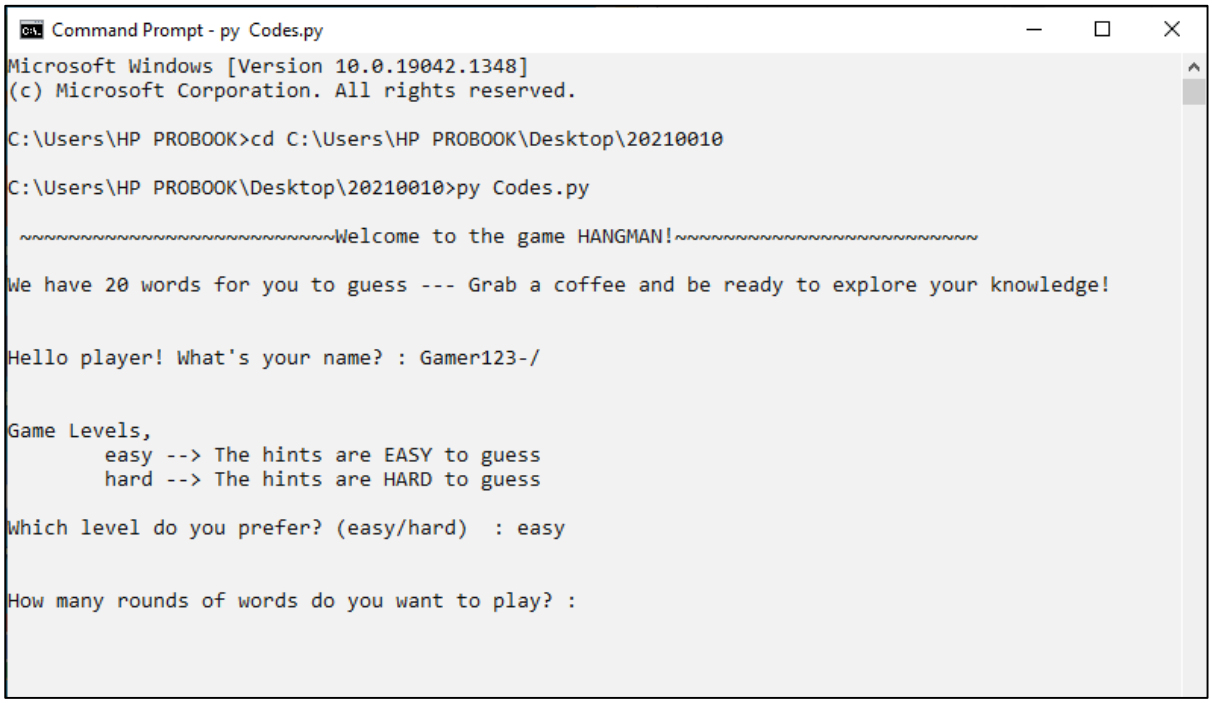


*Figure 3 – Test Case #3 Inserting "easy"*

## **Test Case #4** – *Inserting any character other than "easy" and "hard"*

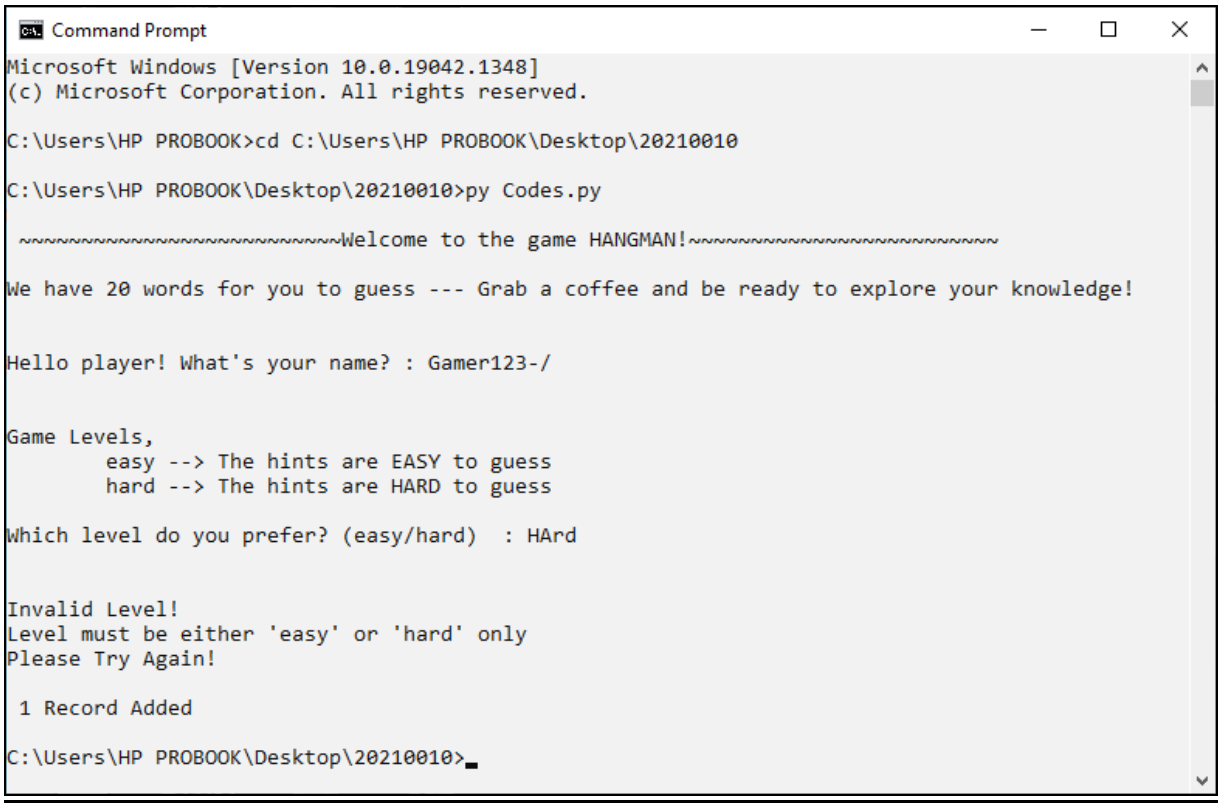


*Figure 4 – Test Case #4 Inserting any character other than "easy" and "hard"*

## Test Case #5 – *Inserting any number which is greater than zero*

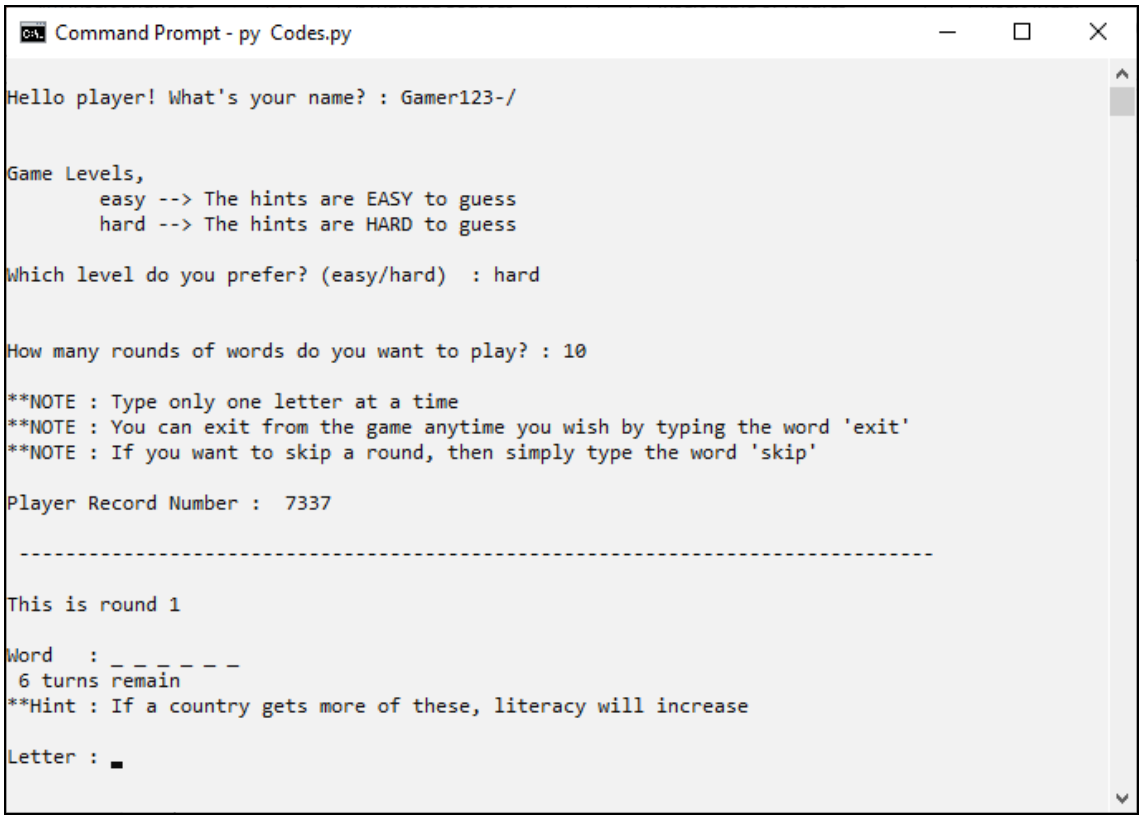


*Figure 5 – Test Case #5 Inserting any number which is greater than zero*

## Test Case #6 – *Inserting any number which is equal or less than zero*

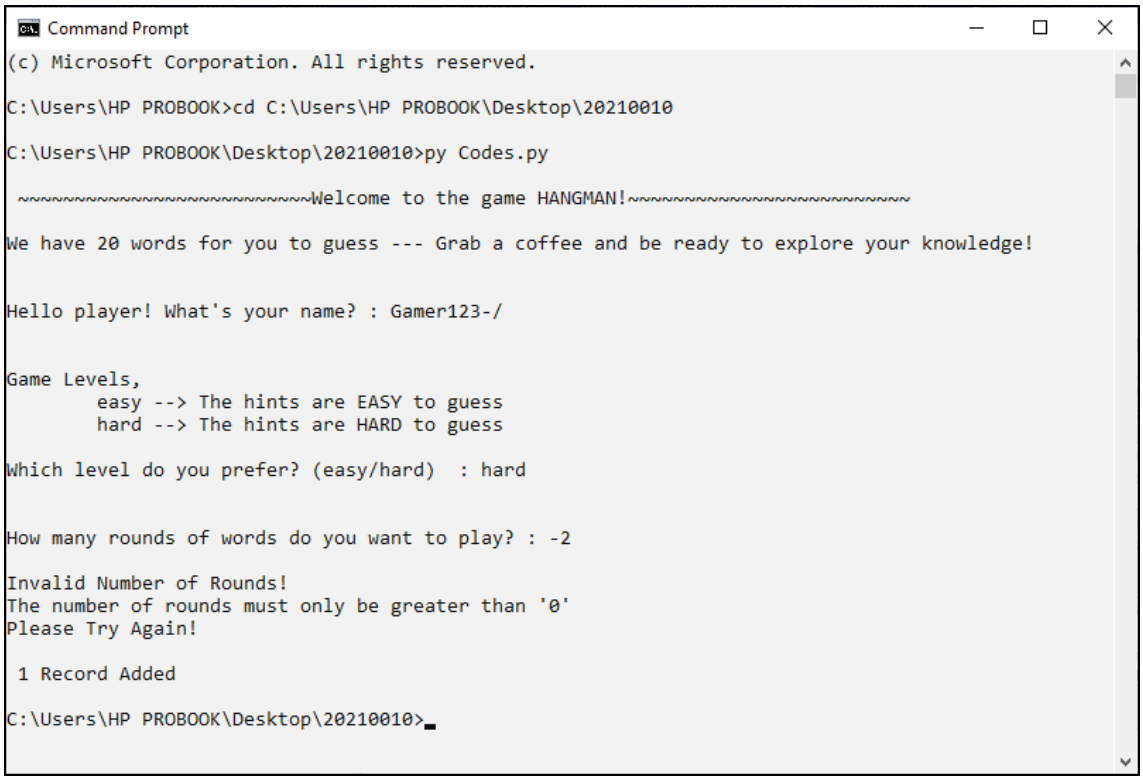


*Figure 6 – Test Case #6 Inserting any number which is equal or less than zero*

## Test Case #7 – *Inserting any letter which is in the word*

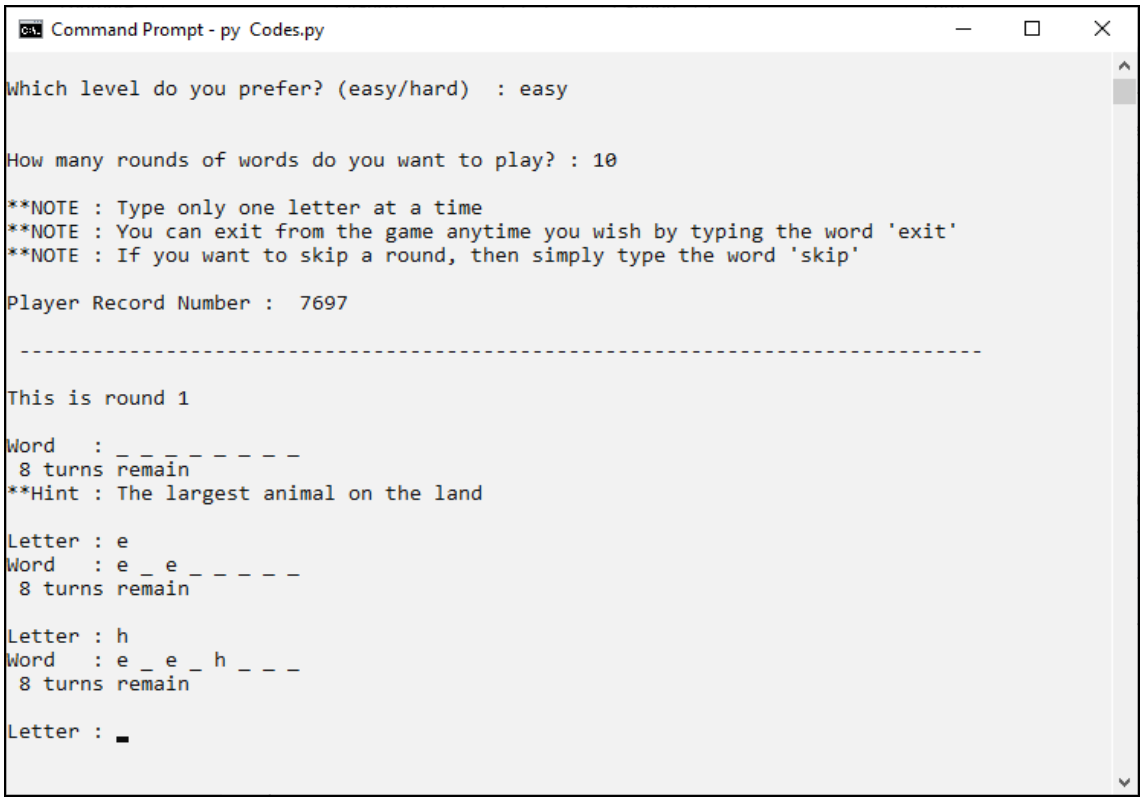


*Figure 7 – Test Case #7 Inserting any letter which is in the word*

## Test Case #8 – *Inserting any letter which is not in the word*

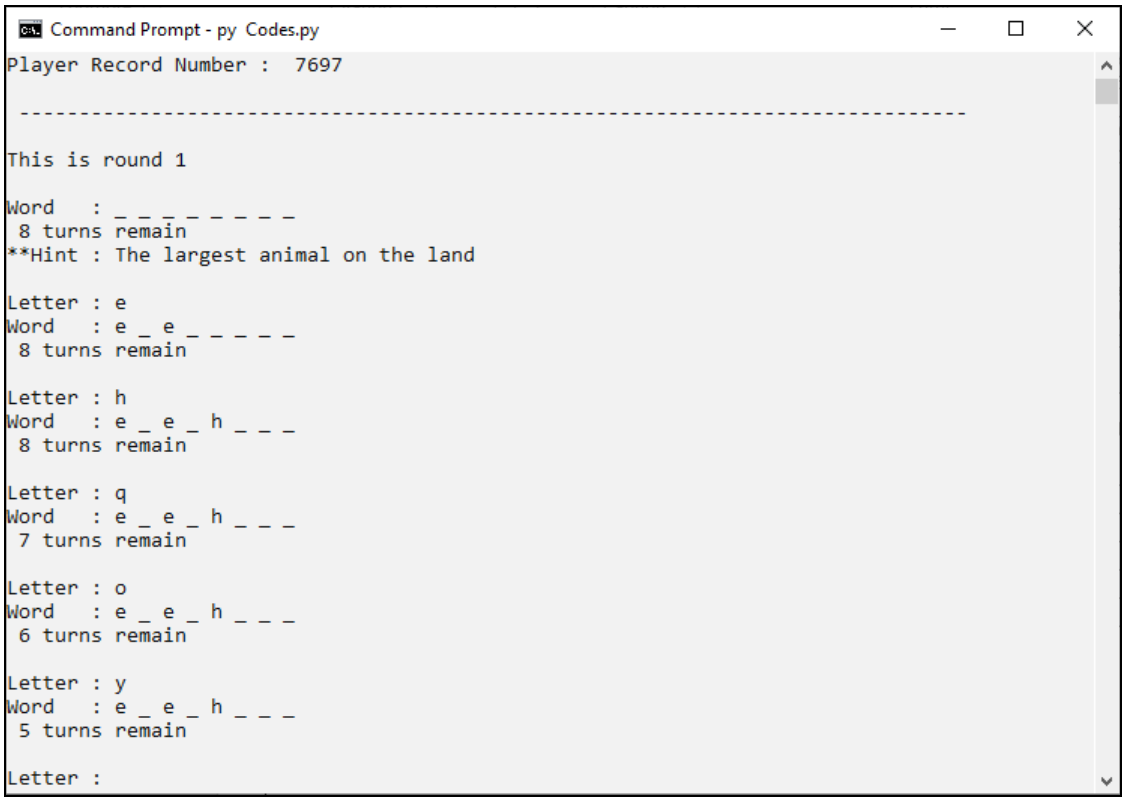


*Figure 8 – Test Case #8 Inserting any letter which is not in the word*

## Test Case #9 – *Inserting a number*

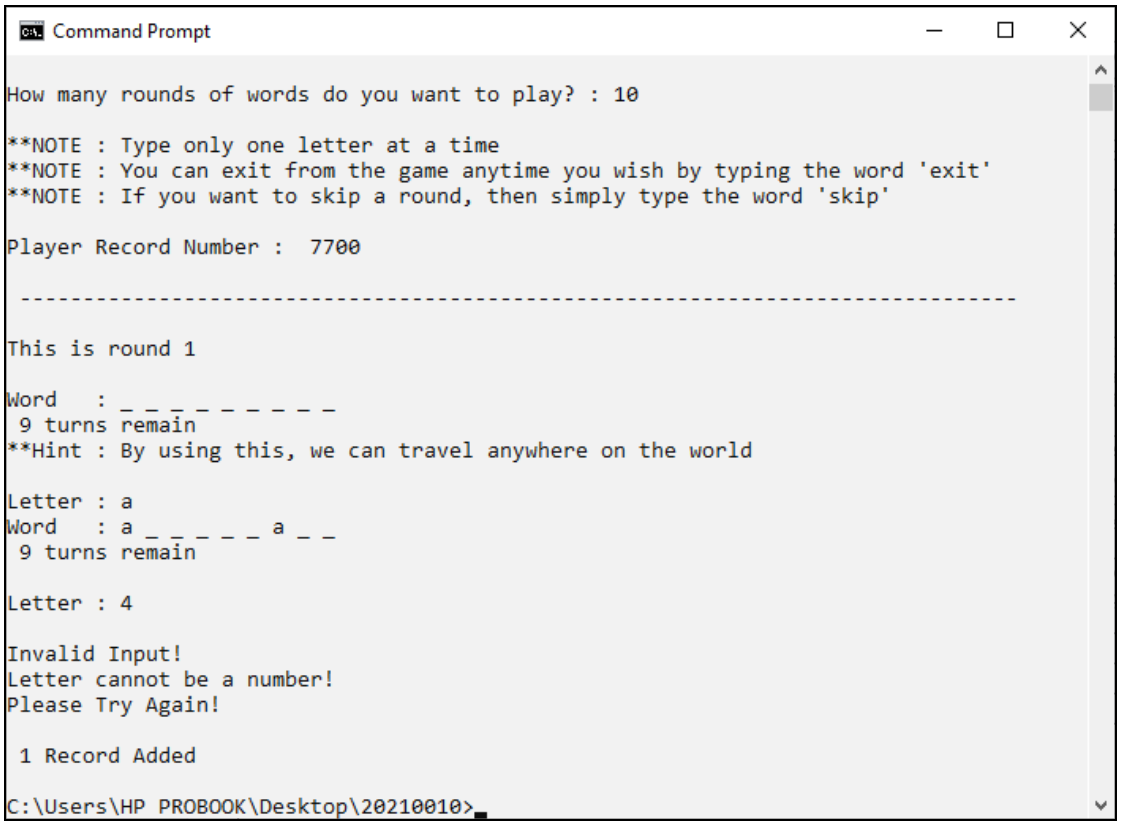


*Figure 9 – Test Case#9 Inserting a number*

## Test Case #10 – *Inserting the letters which are in the word*

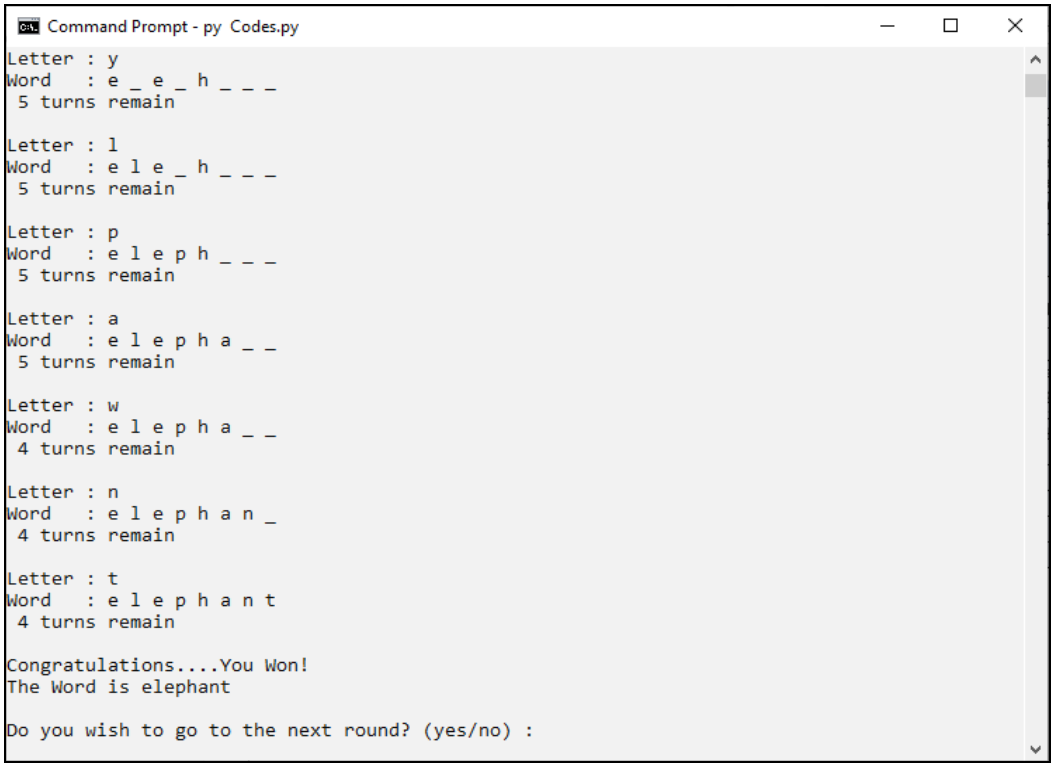


*Figure 10 – Test Case#10 Inserting the letters which are in the word*

## Test Case #11 – *Inserting the letters which are not in the word*

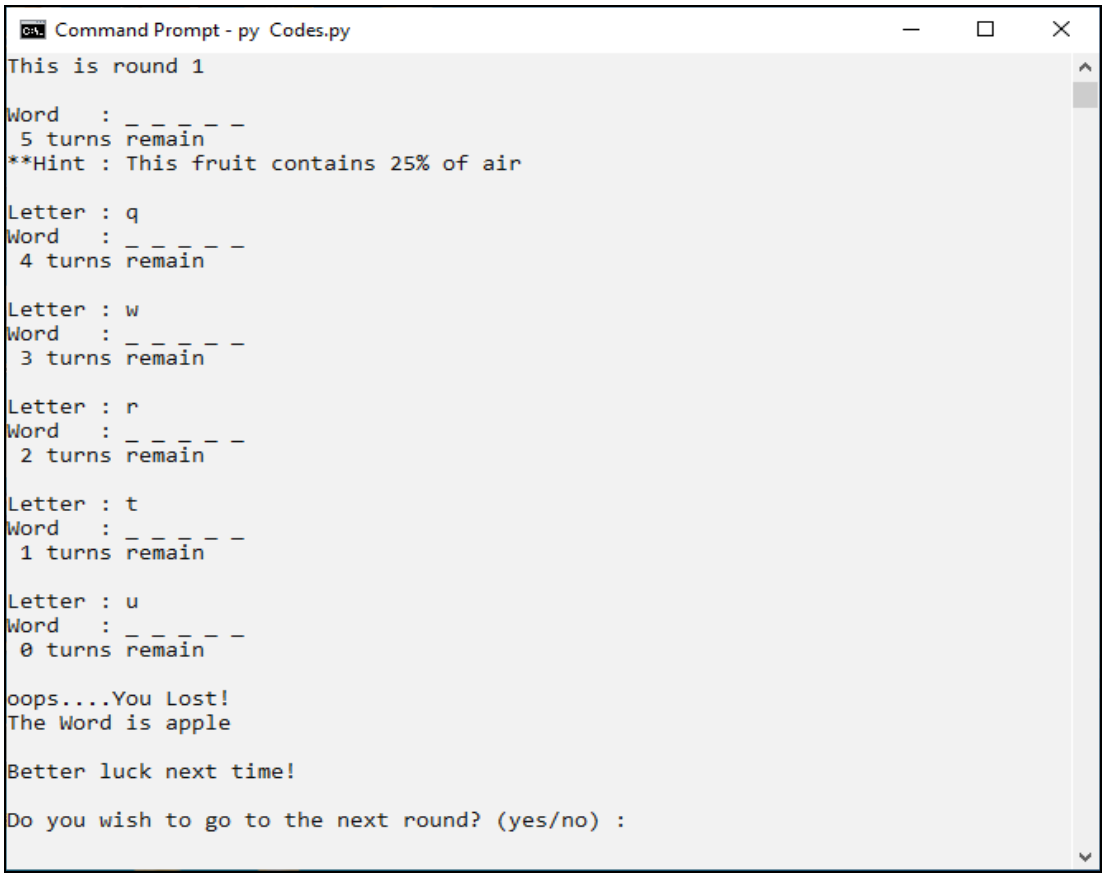


*Figure 11 – Test Case #11 Inserting the letters which are not in the word*

## Test Case #12 – *Inserting "yes"*

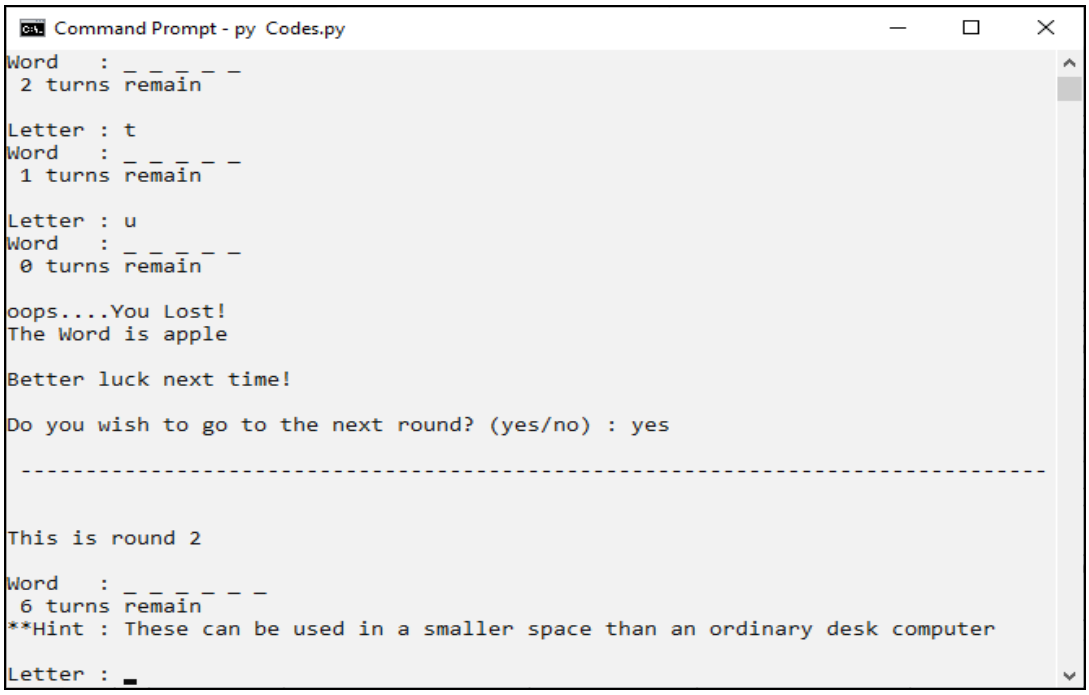


*Figure 12 – Test Case #12 Inserting "yes"*

## Test Case #13 – *Inserting "no"*

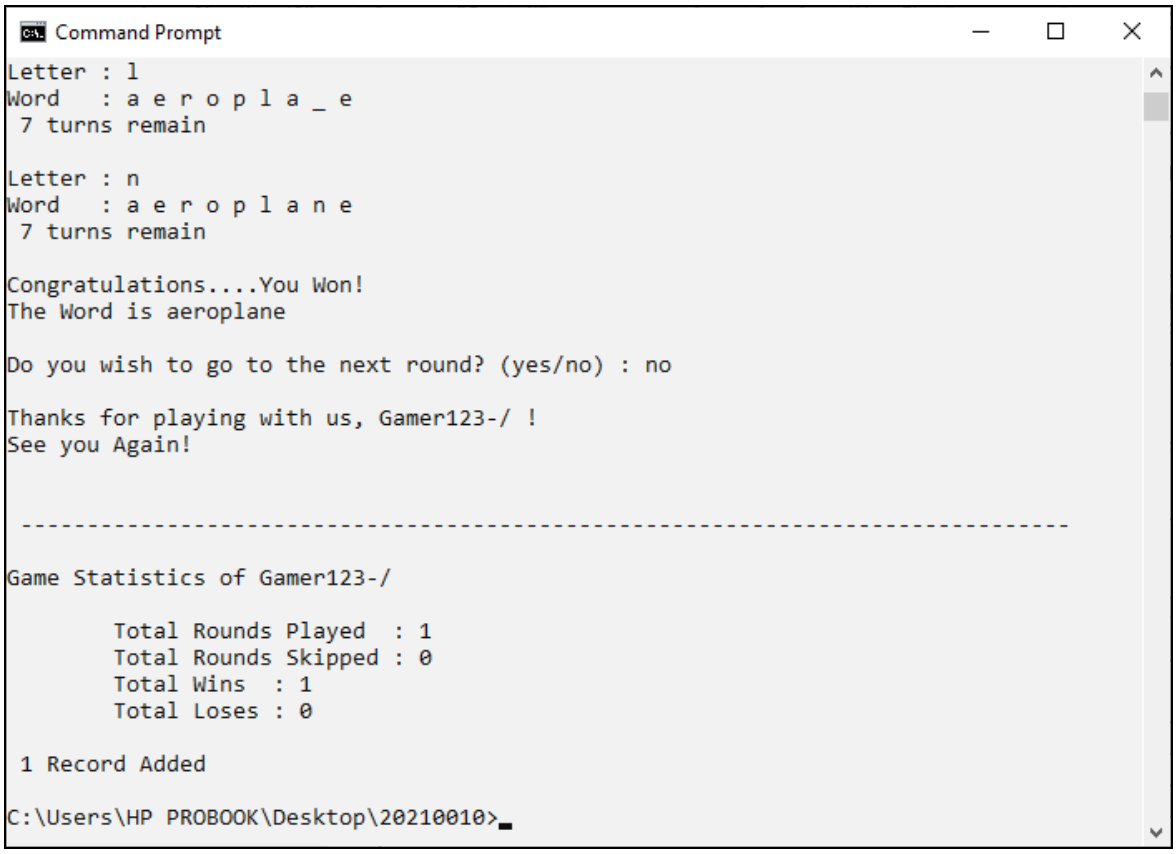


*Figure 13 – Test Case #13 Inserting "no"*

## Test Case #14 – *Inserting any word except "yes" and "no"*

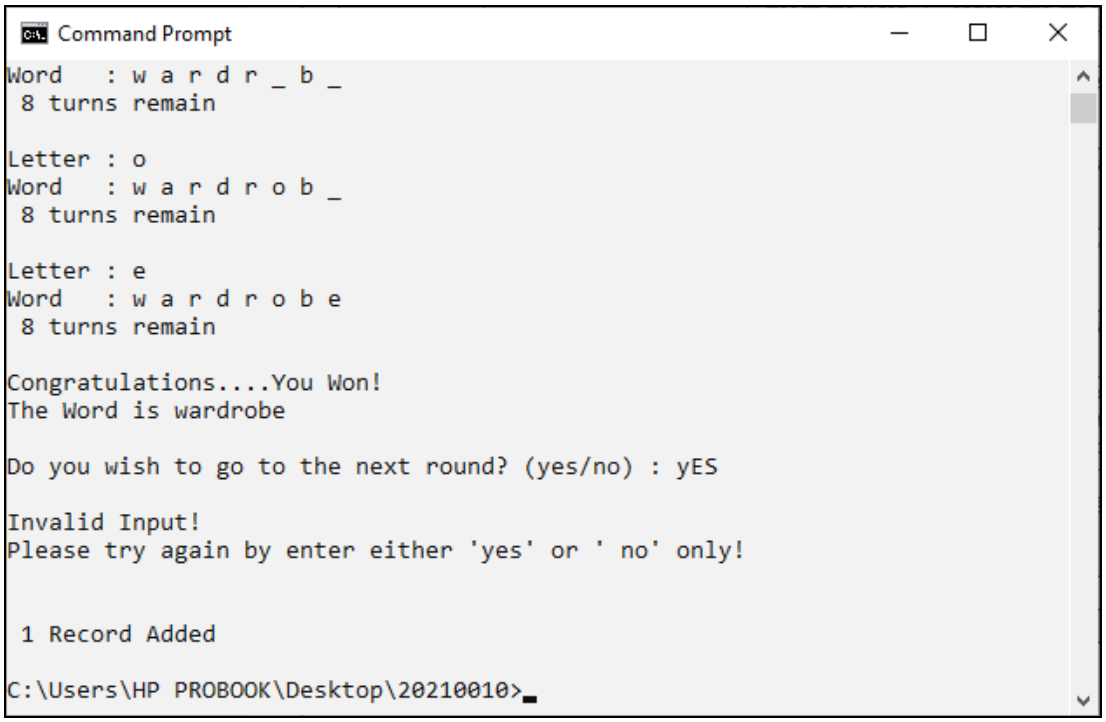


*Figure 14 – Test Case #14 Inserting any word except "yes" and "no"*

## Test Case #15 – *Inserting "skip"*

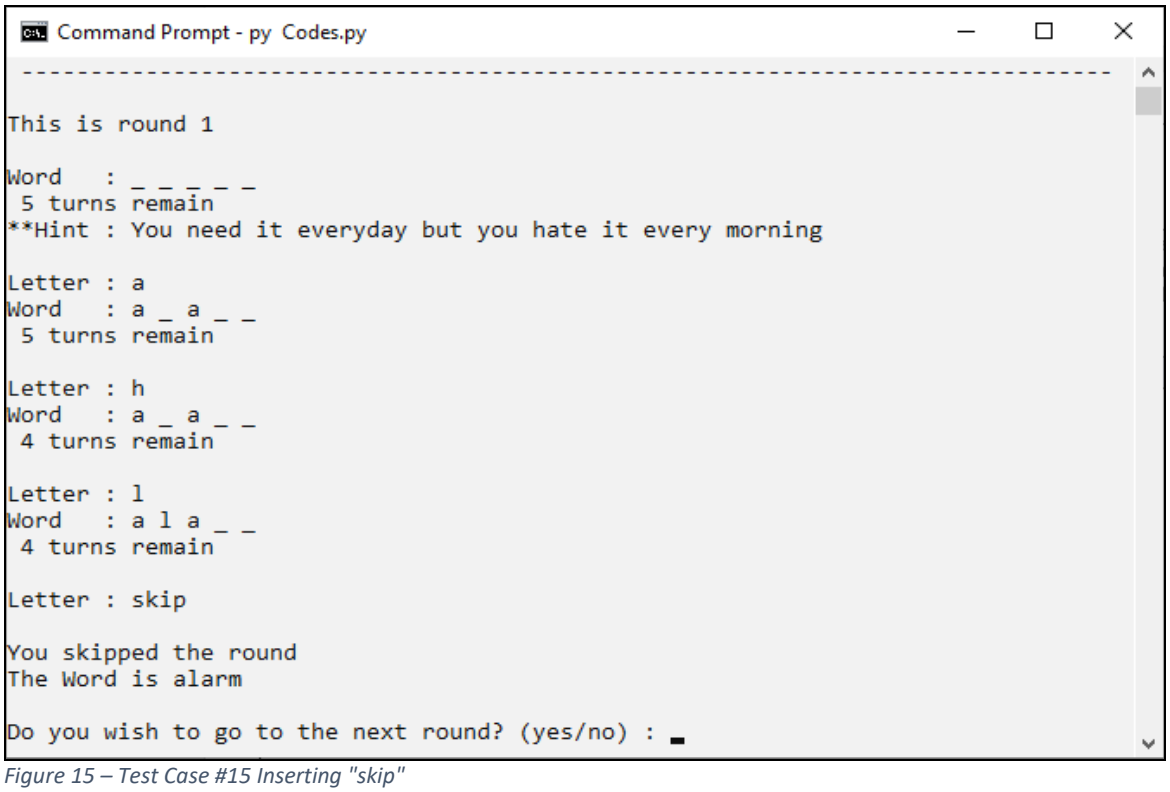


*Figure 15 – Test Case #15 Inserting "skip"*

## Test Case #16 – *Inserting "exit"*


*Figure 16 – Test Case #16 Inserting "exit"*

## 6.6. Screenshot of the HTML file of a session

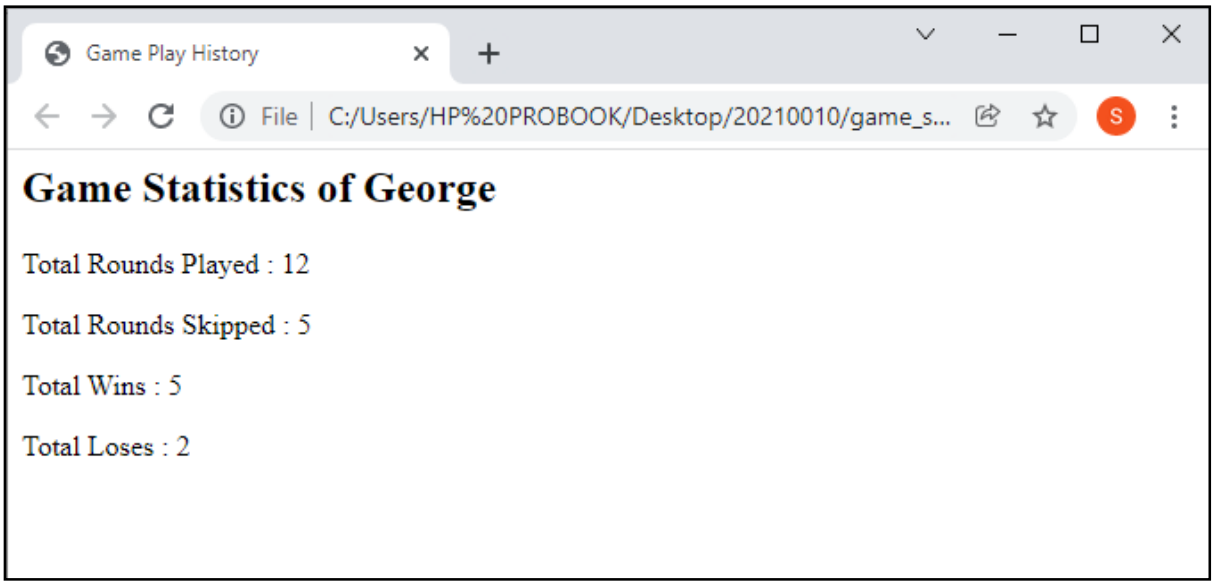


*Figure 17 - Screenshot of the HTML file of a session*

## 6.7. Screenshot of the Database

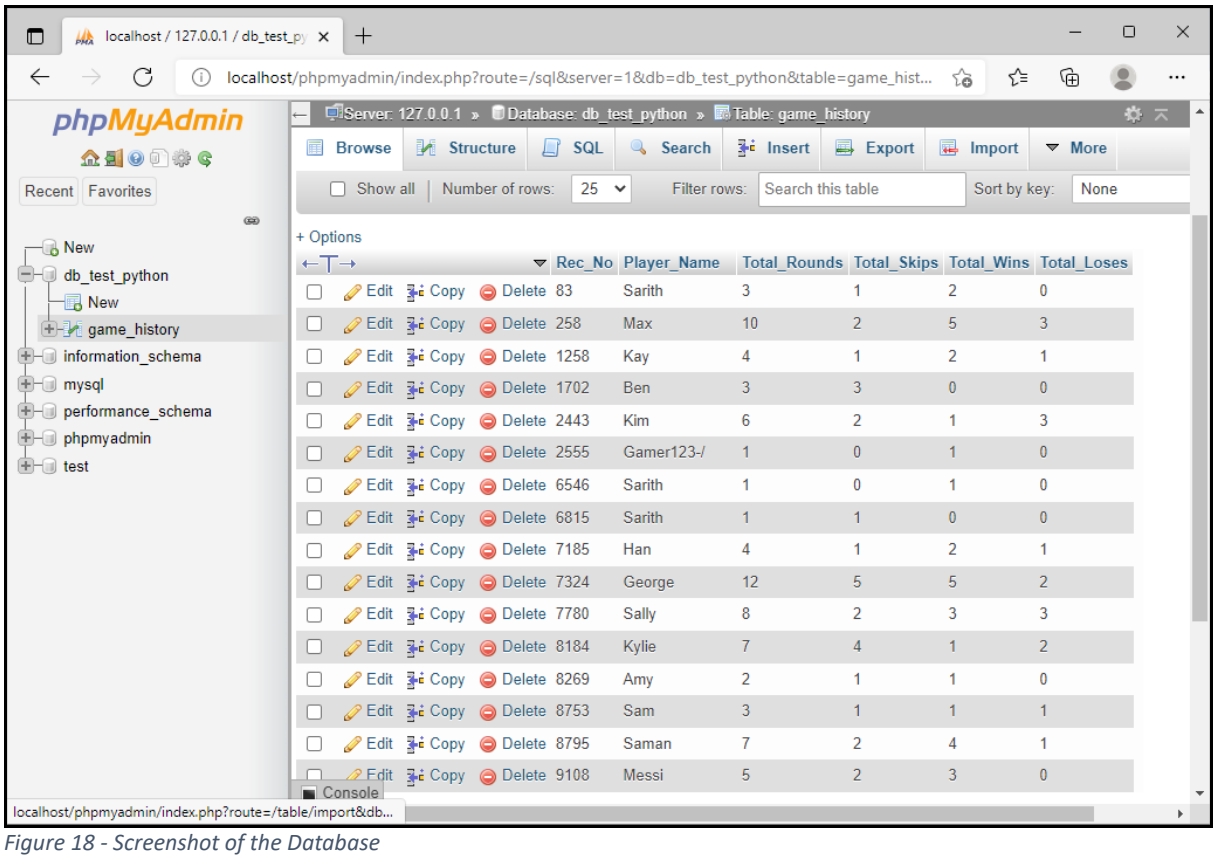


*Figure 18 - Screenshot of the Database*

# 7. Conclusion

This coursework report for the “DOC 334: Introduction to Programming 2” module has done and compiled by 20210010 Sarith. This report begins with a title page including all the requirements. Then there’s an abstract for the readers to get a brief knowledge about the contents including in the report. An acknowledgment is also in the report to express the gratitude for everyone who helped me throughout the coursework. After that, this report contains with the table of contents, the list of tables, and the list of figures to reflect the order of contents to the reader. The main body of the report begins after that. The body part of the report includes a description of the problem statement, a description of the solution I have developed, the program codes as a text, the table of test cases, the screenshots of the test case results or the program outputs, a screenshot of the HTML file of a session, and a screenshot of the database. This report comes to an end with a conclusion which has a summary of the report.