

**Informatics Institute of Technology**

Foundation Program in Higher Education

Module: DOC334 – Introduction to Programming II

Module Leader: Dr. Damitha Karunaratne

Assessment Type: Individual Coursework

Date of Submission: 10/04/2020

Group: C2

Student’s Information: D.G.A Kavinda – 2019149 (avishka.2019149@iit.ac.lk)



**Mathtrix**

A Simple Math Game using Python

# Acknowledgement

First, I would like to thank Dr. Damitha Karunaratne our module leader of Programming Module for guiding and motivating me to complete this project on “Simple Math game using Python” and Mr. Nishan Saliya for showing me the recommended methods on how to complete my coursework and the report. And finally, my classmates, parents and the online community who were a great help throughout the entire project.

Table of Contents

[Acknowledgement 2](#_Toc37335947)

[**Table of figures** 4](#_Toc37335948)

[1.0 Introduction 5](#_Toc37335949)

[2.0 Screenshot of the displayed outputs 6](#_Toc37335950)

[A bit about the Code 11](#_Toc37335951)

[1. Main script 12](#_Toc37335952)

[2. Functions 13](#_Toc37335953)

[3. Database 17](#_Toc37335954)

[4. Utilities 21](#_Toc37335955)

[5. Test Case 22](#_Toc37335956)

[6. References 23](#_Toc37335957)

[‌ 23](#_Toc37335958)

**Table of figures**

[Figure 1- Main menu of the game 6](#_Toc37335800)

[Figure 2- Requesting the user to input his/her name 6](#_Toc37335801)

[Figure 3- Quick game option and the questions 6](#_Toc37335802)

[Figure 4- Post game results screen 7](#_Toc37335803)

[Figure 5- Custom game option - Request Difficulty and Questions 7](#_Toc37335804)

[Figure 6- Custom game option – Questions 8](#_Toc37335805)

[Figure 7- Post game results for custom game 8](#_Toc37335806)

[Figure 8- High scores table 9](#_Toc37335807)

[Figure 9- Error Display 9](#_Toc37335808)

[Figure 10- Same player name identified screen 10](#_Toc37335809)

# 1.0 Introduction

This is a project based on a simple Mathematics game that is built by Python programming language and the player’s scores are stored in a MySQL database.

In this program there are two game modes. Quick game mode and a Custom game mode respectively.

In the quick mode, the player is requested to enter his/her name and given 5 simple additions questions containing operands between 1-10.

In the custom mode, the player is requested to enter his/her name, the difficulty level and the number of questions the player wish to answer. The maximum operand range vary from level to level.

This project has been an educational as well an exciting experience for me and I hope u enjoy it too.

# 2.0 Screenshot of the displayed outputs

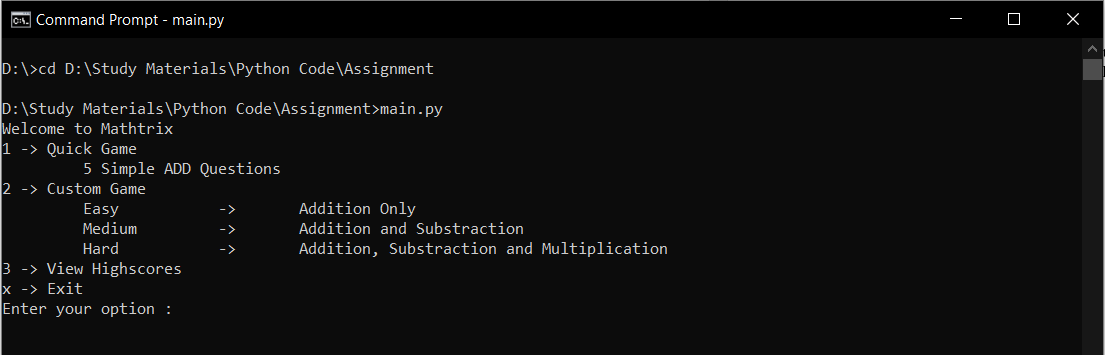


Figure 1- Main menu of the game



Figure 2- Requesting the user to input his/her name

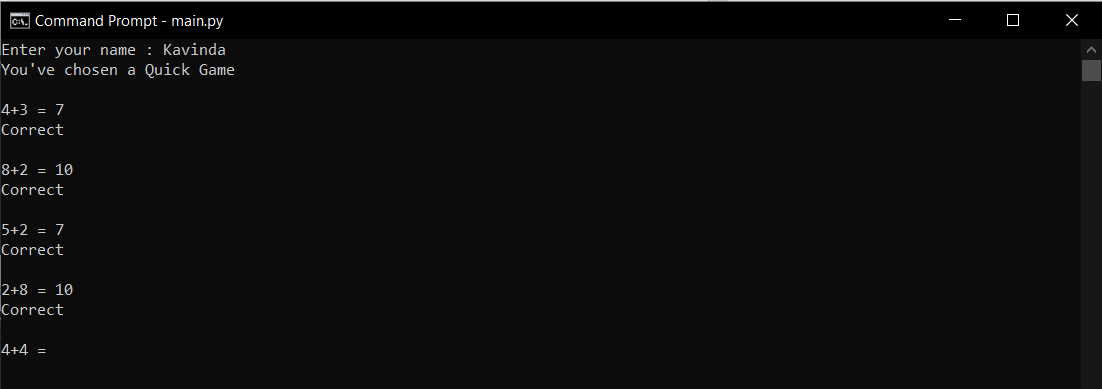


Figure 3- Quick game option and the questions

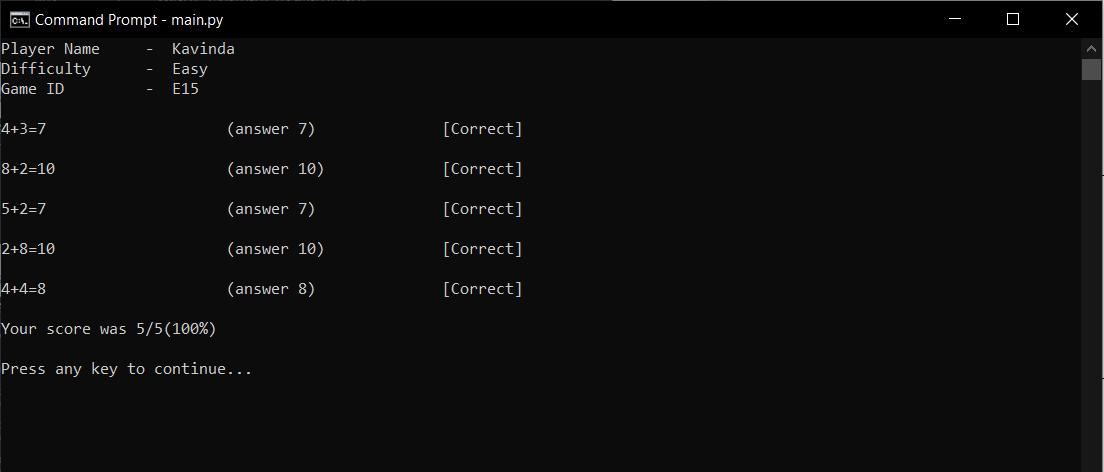


Figure 4- Post game results screen

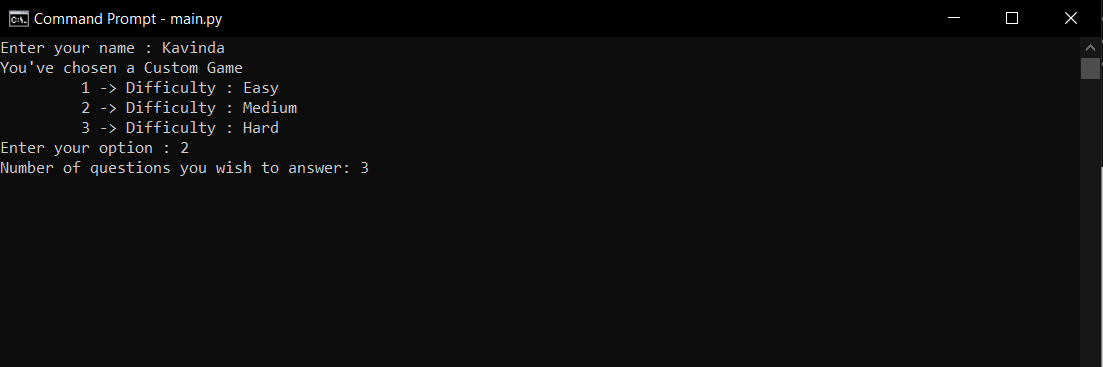


Figure 5- Custom game option - Request Difficulty and Questions

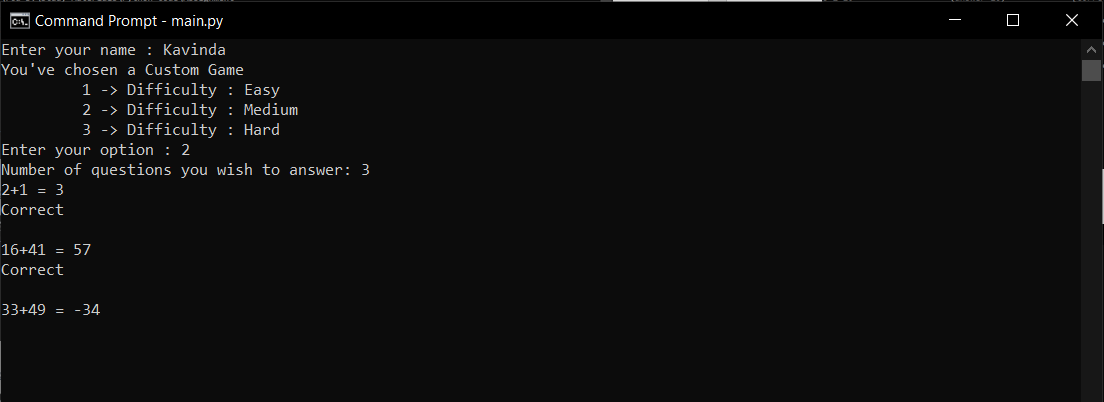


Figure 6- Custom game option – Questions

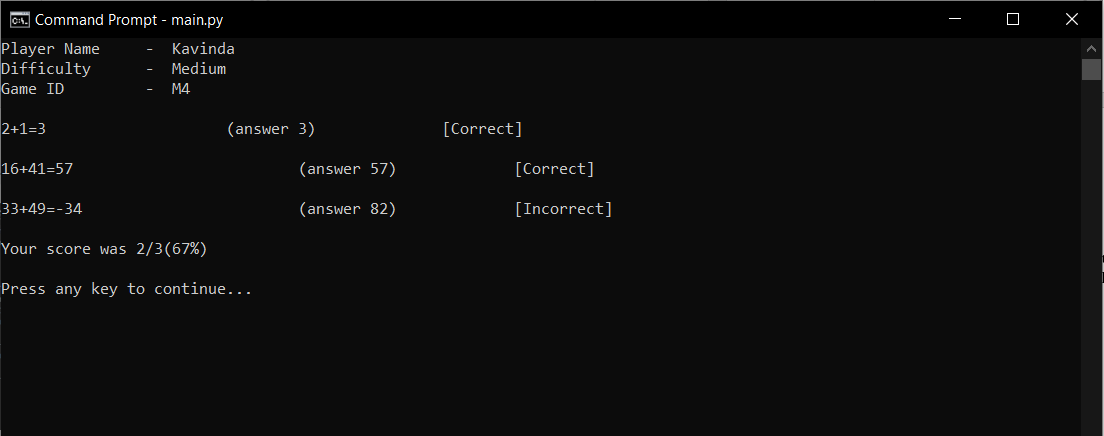


Figure 7- Post game results for custom game

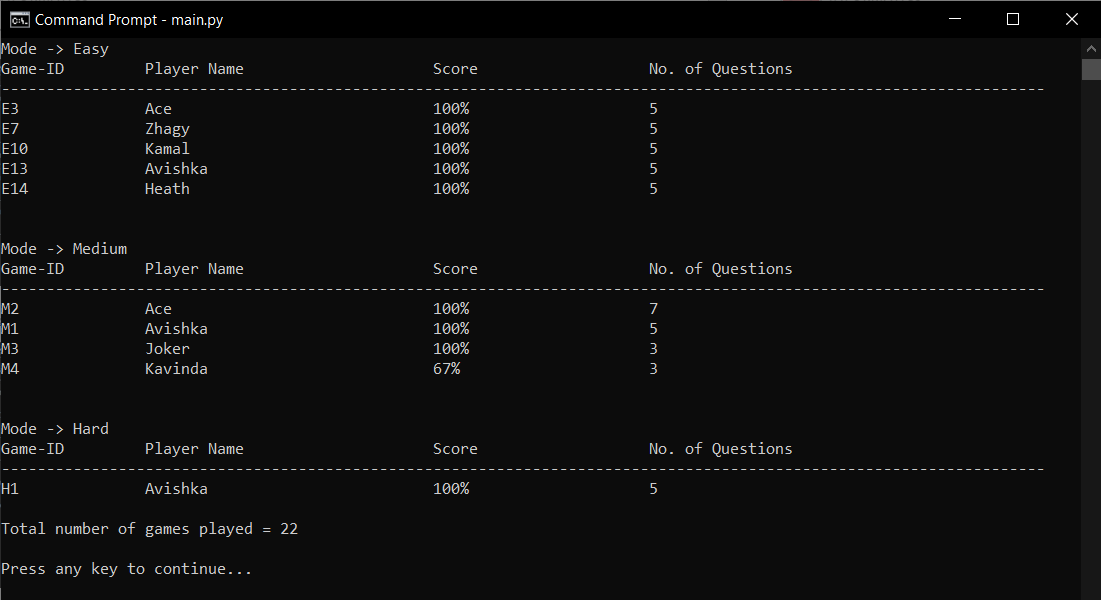


Figure 8- High scores table

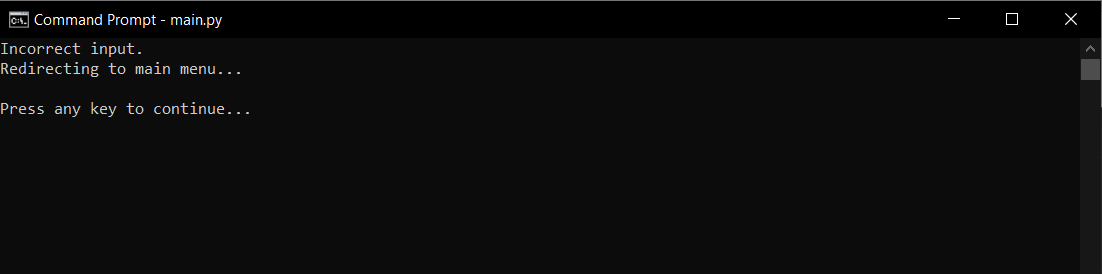


Figure 9- Error Display

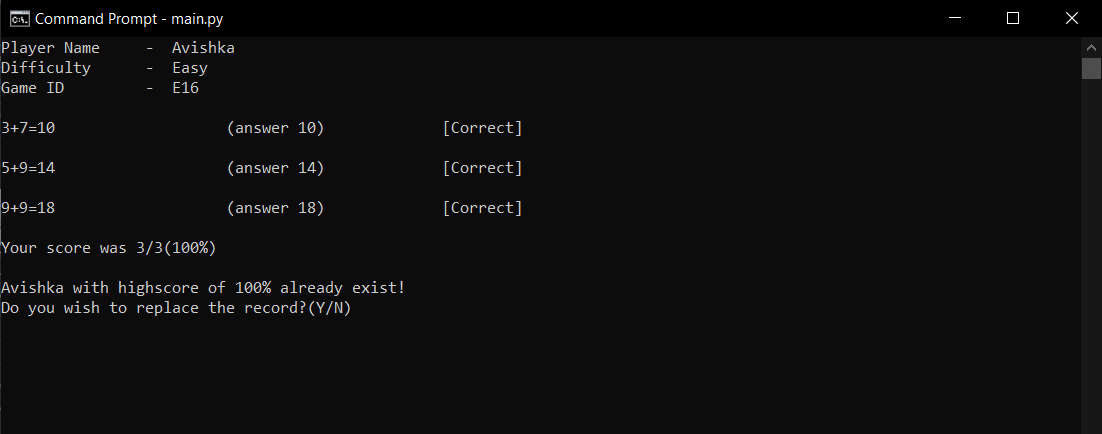


Figure 10- Same player name identified screen

# A bit about the Code

The code of this program can be mainly divided into 4 main sections.

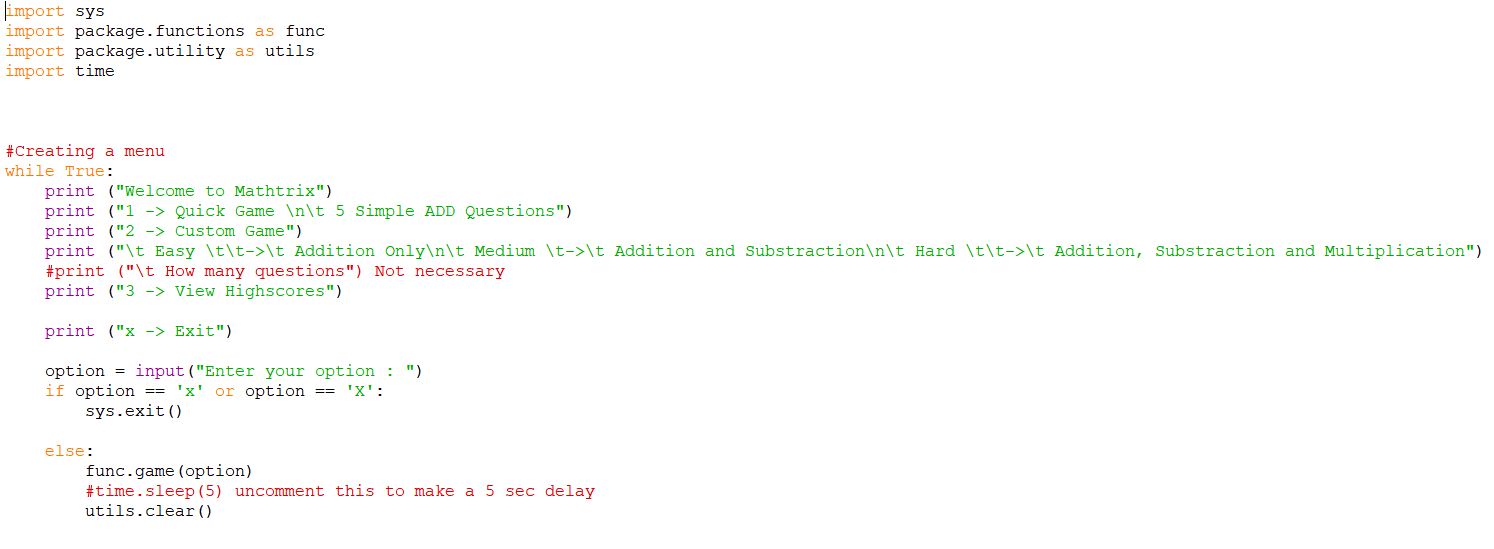
First one being the main script which calls the game() function from the function’s module.

The second is the function module which contains the main functions of the math program, game mode menus and the post-game results display.

The third section is the database handling functions. This module communicates with the database to save player scores to the database and to view player high scores from the main menu.

The fourth section helps the code in using some simple utility functions.

# Main script

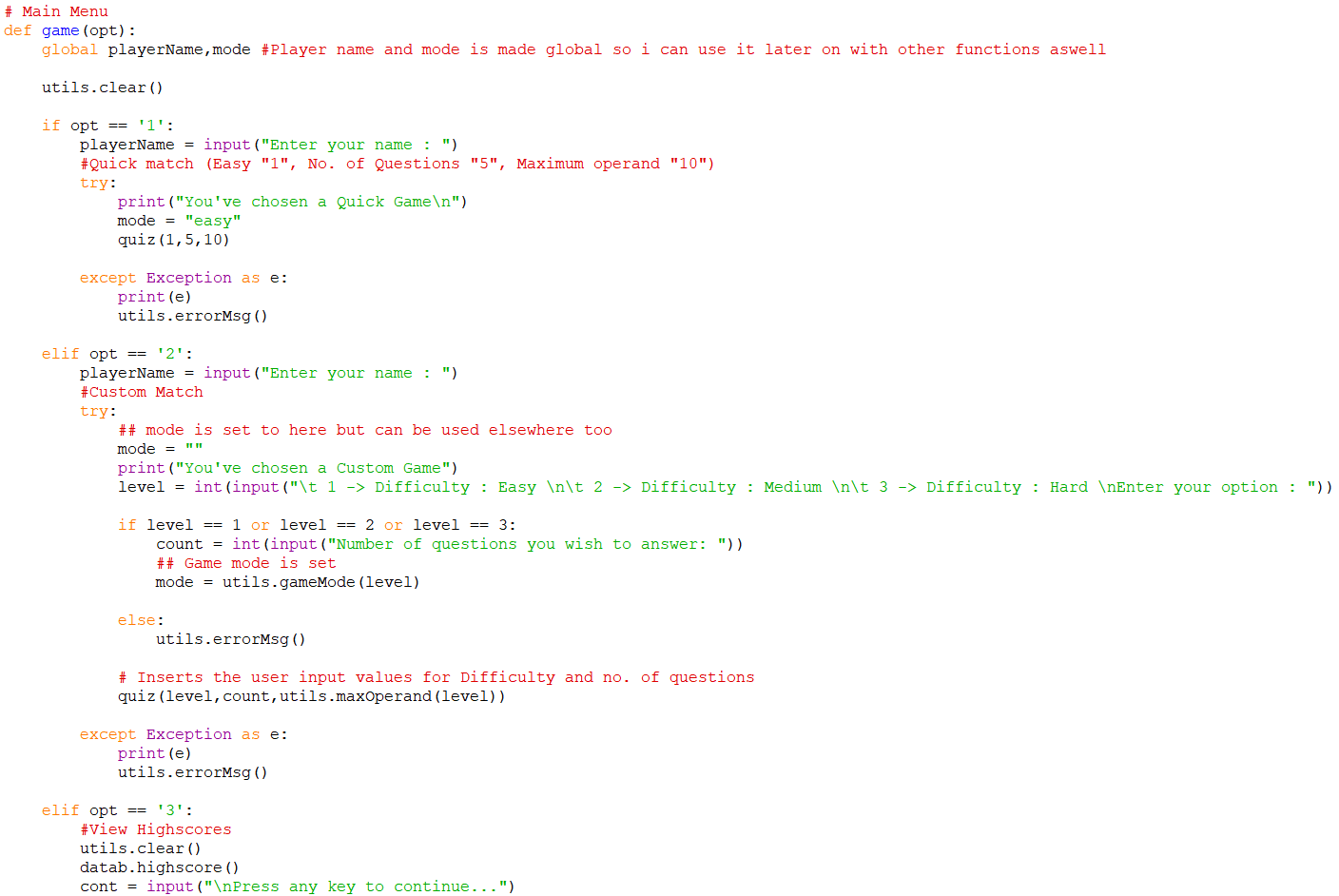


This script displays the main menu of the program.

Exits the program if the user enters ‘x’ or ‘X’ or else runs the game function in the functions module in the package.

And finally clears the command line using the clear() function in utilities module.

# Functions



This is the main function of the game that act as the menu between game modes, difficulty levels and viewing high scores.

\*Player’s name (playerName) and the Difficulty level (mode) is set to global at the beginning of the function, so once set they can be used throughout the functions script.

In try except exception handlers, the exception is printed in the python idle but when run in command prompt it is replaced from the error handling function in utilities. This method is used later in other functions too.

The code is optimized to run quick mode and the custom mode within the same function by passing the difficulty level, number of questions and the operand range to the quiz() function.

In the custom game mode, the difficulty level is requested to be enter as 1,2,3 as Easy,Medium,Hard respectively.

To set mode as Easy Medium Hard with their respective numbers the gameMode() function in utilities module is used by passing the user entered level parameter.

Another function is used to calculate the max operand that is used in the game according to the difficulty level. maxOperand() function in the utilities module is used in this. User entered level parameter is used in here as well.



This function displays all the random generated questions and then put them into a list called gameResult[] with the status of the questions (Correct or Incorrect) and the correct answers.

\*The list gameResult[] and the Score of the current game is set to global, so it can be used while displaying the results after the game session.

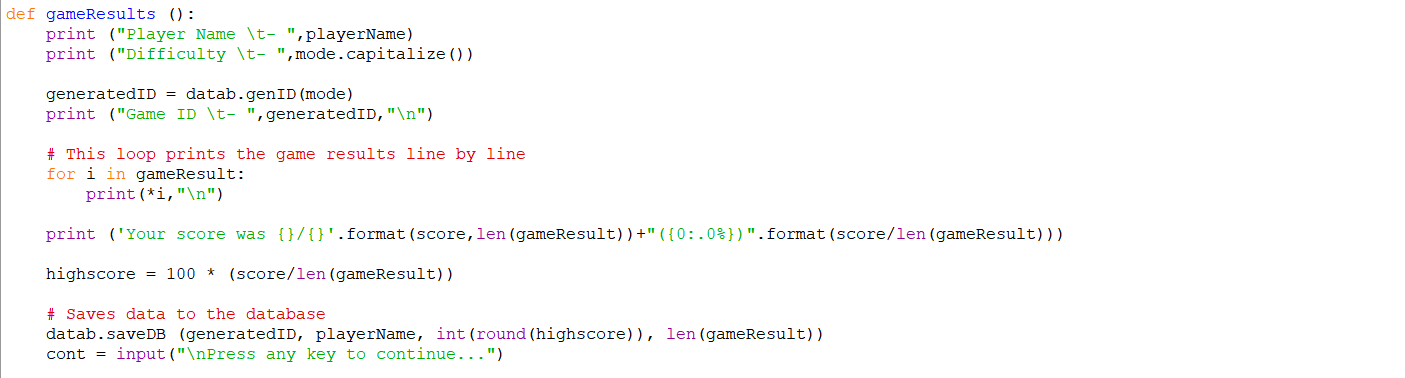
A for loop with the range of count (How many questions player wish to answer or 5 questions in the Quick mode) is used to generate and display the questions.

The two random numbers and the random operator that is selected from the ops[] list respective to the difficulty level is then put into an eval statement to generate the question.

Then the user is requested to answer the question and user’s answer is compared to the result of the eval and the result is printed as correct or incorrect.

Then the question with the users answer, the actual answer and the result is appended to the gameResult[] list at the end of the loop and run until the loop meets the count.

Later the command line is cleared, and the Post game results are displayed using the gameResults() function.



This function displays the post-game results to the player and the score the player achieved.

Player name, Difficulty level, Generated ID for the game is displayed at the top of the screen.

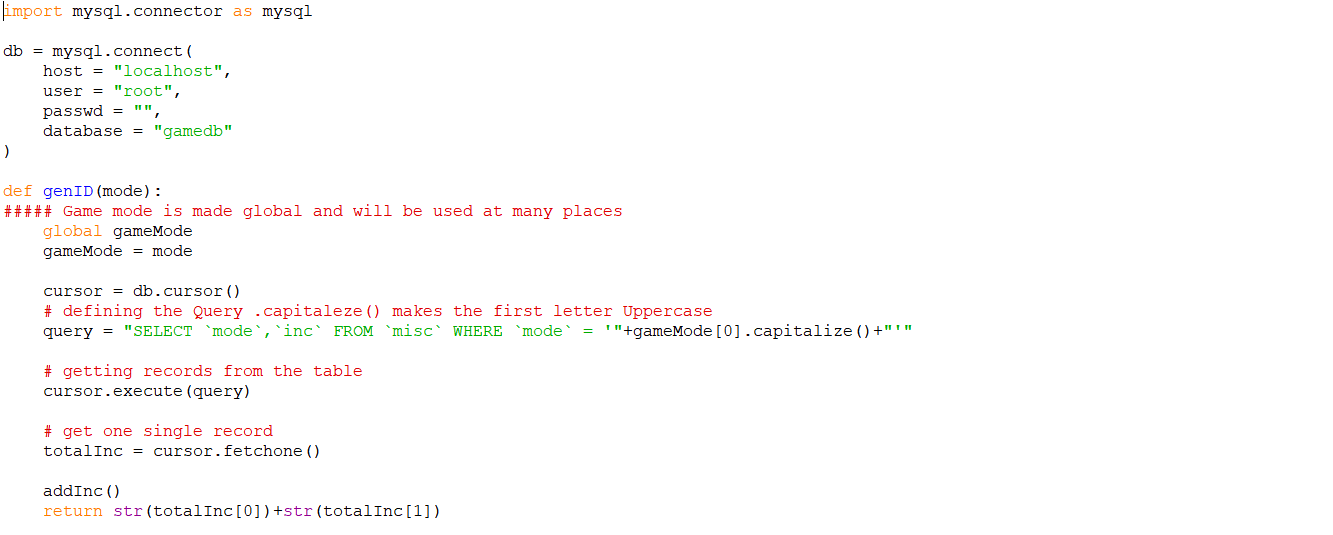
The ID is custom generated in the database module using the genID() function respective to difficulty level.

Then the gameResult[] list is printed inside a for loop.

Then the score the player achieved is displayed.

Then the game ID, Player name, score and the number of questions is saved to the database using the saveDB() function in the database module.

# Database



First, the mySQL connector is imported. It acts like a bridge between the code and the mySQL database.

Then the sql connection is set to a variable named db

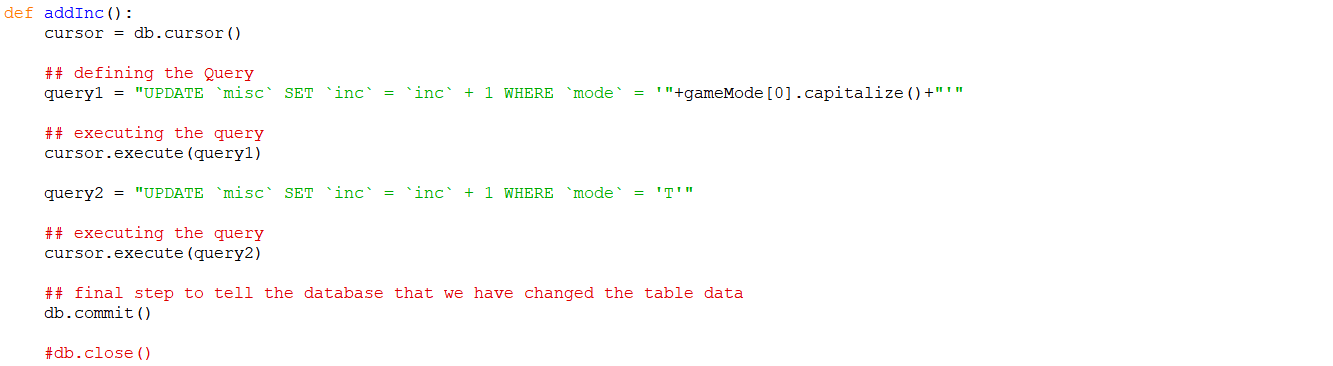
As mentioned before genID() function is used to generate a game ID for the game session. In doing this the game mode is passed from the gameResults() function in the functions module.

\*The gameMode is set to global because it is used throughout the script.

Then the corresponding values are selected from the ‘misc’ table in the database ‘gamedb’.

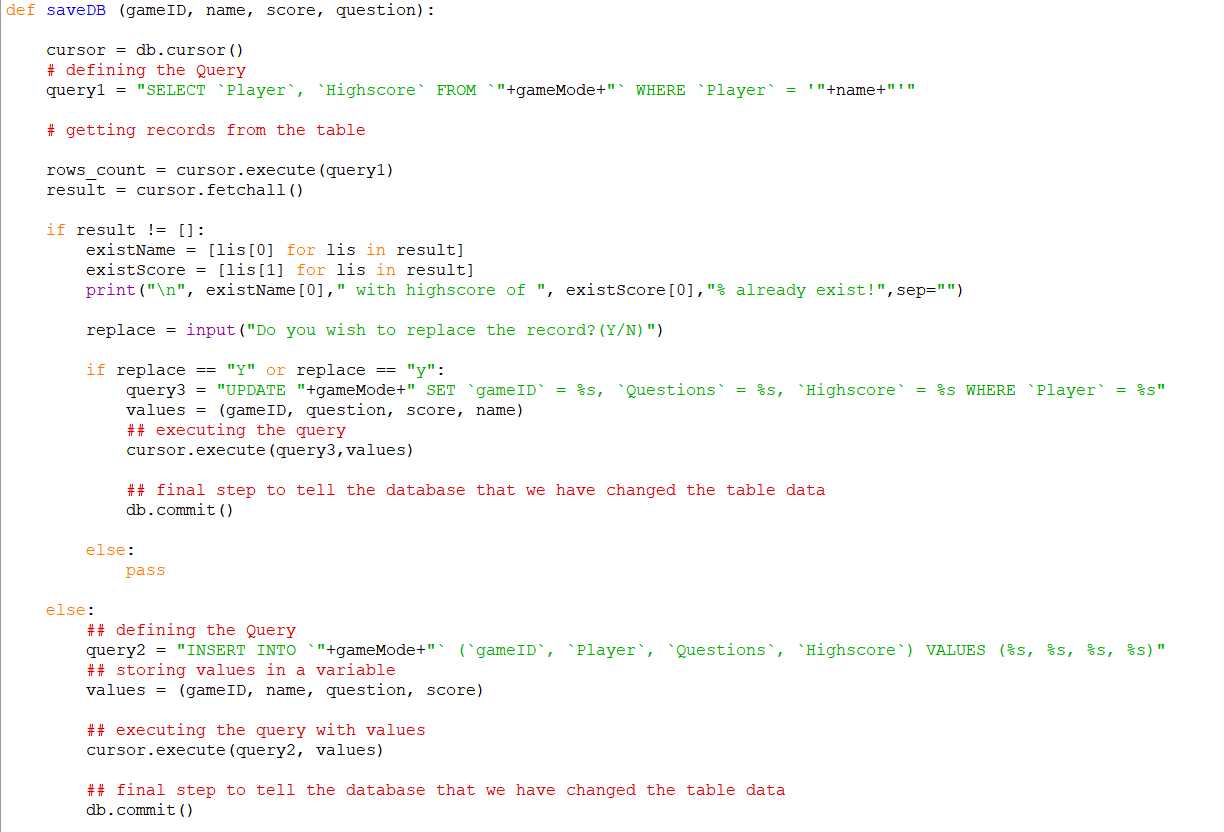
And the increment is increased by 1 using the addInc() function.

Finally, the generated ID is returned to the gameResult() function in functions module.



This function is used to increase the increment of the respective modes in the ‘misc’ table.

And this function also increases the amount of total games played.

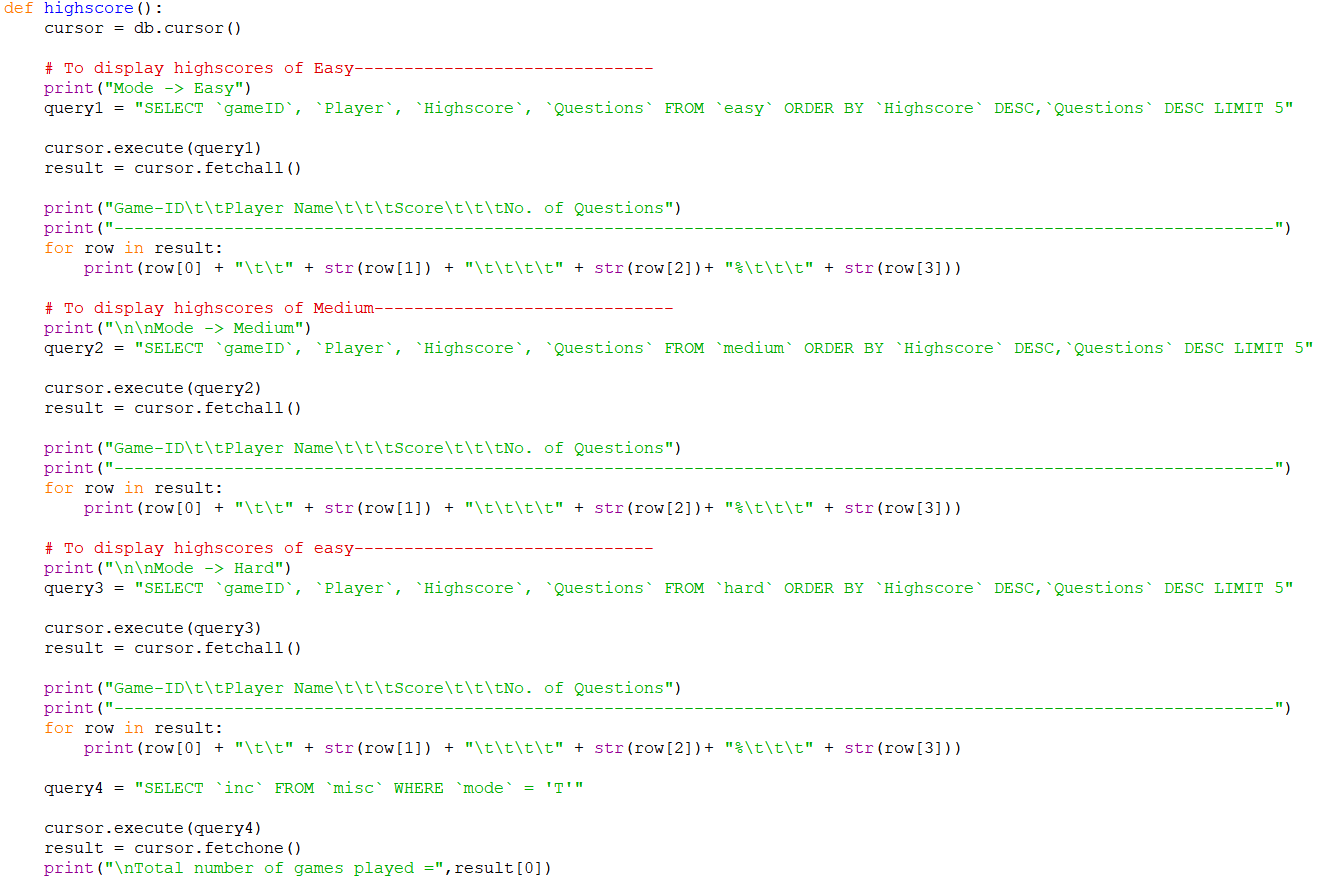


This function saves the game data to the database using the parameters passed from the functions table and to the respective tables using the gameMode variable.

This function first checks if there’s an existing player with the same name as the player entered.

If the result is not null or empty, the user is asked whether he/she wish to replace the existing record or no. If the player agrees to replace the existing data, the ‘UPDATE’ query is executed.

If the result is null or empty, the user is inserted into the database as a new user using the ‘INSERT’ query.



This function displays the high scores when requested from the main menu of the program.

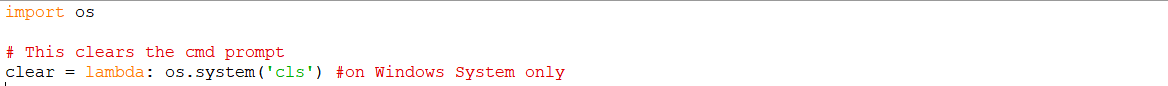
This function selects the values of the easy, medium, hard tables from the database and displays them inside of for loops respectively.

Records that are printed is limited to 5 so that in which they are sorted in descending order by the values of ‘Highscores’ and ‘Questions’ columns of the tables.

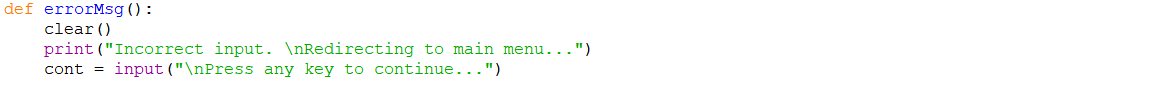
And finally the total of number of games are displayed using the ‘misc’ table.

# Utilities

These functions help the main functions to run properly



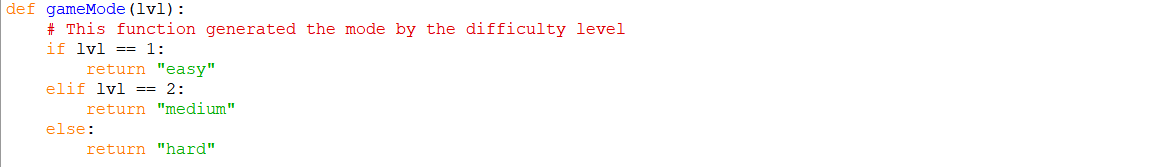
This function can be called by clear() and it clears the command prompt.



This function acts as an error handler. It clears the prompt and print the Incorrect input and requests user to Enter any key to continue.



This function returns the maximum operand respective to the level of difficulty chosen by the user.



This function returns the literal value (Easy, Medium, Hard) respective to the user input difficulty level.

# Test Case

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Case ID | Test Scenario | Test Steps | Test Data | Expected Result | Actual Result | Pass/Fail |
| TU01 | Check “Quick Game” mode | 1.Select “Quick game” mode by entering number “1”  2.Enter Player’s Name | Choice = 1  Player’s Name = Avishka | Player should get 5 simple addition questions | Expected result | Pass |
| TU02 | Check “Custom Game” mode | 1.Select “Custom Game” mode  2.Enter Player’s Name  3.Select Difficulty Level  4.Enter How Many Questions Need | Choice = 2  Player’s Name = Avishka  Difficulty Level = Medium  How many Questions = 3 | Player should get questions according to his selections | Expected result.  Player got 3 medium difficulty level questions | Pass |
| TU03 | Check Last Game Scores | 1.Select “Display Last Game Scores By entering Number “3” ” | Choice = 3 | Player should see scores of past game players | Expected result.  Displays the high scores table | Pass |

# References

W3schools.com. 2020. *Python Mysql*. [online] Available at: <https://www.w3schools.com/python/python\_mysql\_getstarted.asp> [Accessed 1 April 2020].

Python For Beginners. 2020. *Exception Handling In Python*. [online] Available at: <https://www.pythonforbeginners.com/error-handling/exception-handling-in-python> [Accessed 1 April 2020].

Tutorialspoint.com. 2020. *Python Mysql - Insert Data - Tutorialspoint*. [online] Available at: <https://www.tutorialspoint.com/python\_data\_access/python\_mysql\_insert\_data.htm> [Accessed 2 April 2020].

GeeksforGeeks. 2020. *Python | Output Formatting - Geeksforgeeks*. [online] Available at: <https://www.geeksforgeeks.org/python-output-formatting/> [Accessed 2 April 2020].

EDUCBA. 2020. *Mysql Query Commands | Concepts | Basic To Advanced Commands*. [online] Available at: <https://www.educba.com/mysql-query-commands/> [Accessed 5 April 2020].

Codecademy. 2020. *List Of SQL Commands | Codecademy*. [online] Available at: <https://www.codecademy.com/articles/sql-commands> [Accessed 6 April 2020].