# Programming Project II

## PROG1700 Intro to Programming : Python

**Evaluation :** 20% of final grade

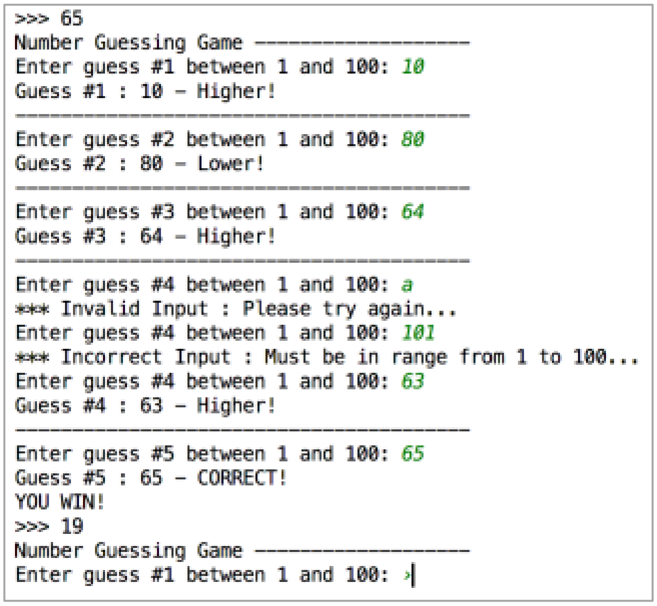
**Due Date :** Nov 20 (Web) / Nov 20 (Systems)

## Project Description

Project II will be to develop a python application number guessing game.

The requirements of the project are:

1. At start up, the application will randomly generate a number between 1 and 100 (the target number). It then prompts the user to enter his / her guess (the guessed number). The prompt must include the turn the user is currently on (the turn number).
2. The user has ten chances to guess the correct number



```
>>> 65
Number Guessing Game --------------------
Enter guess #1 between 1 and 100: 10
Guess #1 : 10 - Higher!
----------------------------------------
Enter guess #2 between 1 and 100: 80
Guess #2 : 80 - Lower!
----------------------------------------
Enter guess #3 between 1 and 100: 64
Guess #3 : 64 - Higher!
----------------------------------------
Enter guess #4 between 1 and 100: a
*** Invalid Input : Please try again...
Enter guess #4 between 1 and 100: 101
*** Incorrect Input : Must be in range from 1 to 100...
Enter guess #4 between 1 and 100: 63
Guess #4 : 63 - Higher!
----------------------------------------
Enter guess #5 between 1 and 100: 65
Guess #5 : 65 - CORRECT!
YOU WIN!
>>> 19
Number Guessing Game --------------------
Enter guess #1 between 1 and 100: ›
```

3. After each guess, a hint will be displayed to the user in the console outlining whether the target number is higher or lower than the guessed number. The output should also include the guessed number and the turn number. Output formatting must be used for the hint. See diagram for details.
4. If the user guesses the correct number a message is displayed in the console that the user is a winner and the game is over.
5. If the user uses up all 10 guesses and still has not guessed the correct number, then a message is displayed in the console that user has lost and the game is over.
6. When the game is over the application will wait two seconds and then restart the game automatically. It will do this forever until the user manually shuts down the program. There is no requirement to program a way to shut down the game. Playing again must be done with a loop – do not re-call the main() function as this causes a recursive loop!
7. The application must error check for the following and display an error message:
   - guesses out of range
   - decimal numbers
   - letters, spaces, symbols
   - user enters the same guess in twice

   Naturally, when any of these errors occur, the entered guess is not taken and the user is prompted again to enter his / her guess. See diagram for details.
8. Besides the main() function, you must incorporate a user function for getting input and one for processing the input and displaying the hint.
9. No global variables are allowed for this project (nor are they required!)
10. Be sure to include informative comments in your code
11. To aid in testing, add an initial print() statement that displays the correct answer in the console at the start of each new game. See diagram for details.

> See other side

## Requirements (Marks breakdown)

| | |
|---|---|
| Application Test Cases | |
| Application test cases<br>*A series of test cases that test all requirements listed above* | 18 |
| Code Requirements and Design | |
| User functions implemented and other code requirements<br>Commenting in Code | 3<br>1 |
| TOTAL MARK | 22 |

## Other Notes & Hints

- This project will be marked through code review on the due date during out lesson time. Please bring the completed python file to the lab for marking.