Beginning Python: Lesson 4

The Number Guessing Game

In this lesson we'll create a number guessing game in Python. The game goes like this: the computer chooses a random number between 1 and 100 and the player tries to guess it in as few attempts as possible. Each time the player enters a guess, the computer tells the player whether the guess is too high, too low, or correct.

1. Planning the game

It's a good idea to plan your programs before you write them. Programming is a lot like building a house. If a builder builds you a house without a plan, you might end up with a house that looks awful, has 12 bathrooms, no windows, a front door on the second floor! Without a plan, your program might not work correctly and might take ages to write. In programming, an *Algorithm* is a set of instructions to guide you along as you code. Good algorithms are easy to follow and do not leave any steps out. For out guessing game, our algorithm might look like this:

```
welcome the player to the game and explain it
pick a random number between 1 and 100
ask the player for a guess
set the number of guesses to 1
while the player's guess does not equal the number
    if the guess is greater than the number
```

```
        tell the player to guess lower
    otherwise
        tell the player to guess higher
    get a new guess from the player
    increase the number of guesses by 1
congratulate the player on guessing the number
let the player know how many guesses it took
```

Now we will code each step in this algorithm

1. Create a new program, `guessing-game.py`

All good programs begin with a block of comments. The comment block says what the program does and who wrote it. The `#` at the start of each line tells Python that this is a comment.
- Open Wing IDE on your computer. Click on the *File* menu and select *New*. A new blank window will appear for us to type our program in.
- Enter the following comments in the editor window and save the file as `guessing-game.py` in the “python-programs” folder you created in the last lesson

```
# Guess My Number
# The computer picks a random number between 1 and 100.
# The player tries to guess it.
# YOUR_NAME 28/2/2015
```

2. Import the Random module

The program needs to generate a random number. Import the `random` module like you did last week. Add the following code to the program

```
import random
```

3. Welcome the player and explain the game

We want to tell the player how the game works. Add the following highlighted lines of code to your program to do this.

```
print("Welcome to 'Guess My Number'!")
print("I'm thinking of a number between 1 and 100.")
print("Try to guess it in as few attempts as possible.")
```

4. Set the initial values

Next, we need to set two variables:
- the random number the person has to guess
- the initial guess of the player
- the number of tries taken
The following code does this for us:

```
number = random.randint(1,100)
print("Take a guess: ")
guess = int(input())
tries = 1
```

`number` represents the number the player has to guess. We assign it a random integer from 1 to 100. Next, `int(input())` gets the guess from the player and converts it to a number. We assign this number to guess. We assign `tries` the value 1 and this represents the number of tries the player has had.

5. Creating a Guessing Loop

This is the main part of the program. The loop executes as long as the player hasn't correctly guessed the computer's number. During the loop, the player's guess is compared to the computer's number. If the guess is
higher than the number, *Lower...* is printed; otherwise, *Higher...* is printed. The player enters the next guess, and we add 1 to tries.

```
while (guess != number):
    if (guess > number):
        print("Lower...")
    else:
        print("Higher...")
    print("Take a guess: ")
    guess = int(input())
    tries = tries + 1
```

6. Winning the game

When the player guesses the number, `guess` is equal to `number`, which means that the loop condition, `guess != number`, is false and the loop ends. At that point, the player needs to be congratulated:

```
print("You guessed it! The number was", number)
print("And it only took you", tries, "tries!")
```

Now, run your program and check it works correctly. See if you can come up with a way to find the number quickly.

7. Challenge

If you've finished this program, you can try the following challenges:

- Modify the Guess My Number game so that the player has 7 guesses. If the player fails to guess in time, the program should display the message "Sorry, you lose!".
- Write a program that flips a coin 100 times and then tells you the number of heads and tails.

Super Challenge!!!
Write a new guessing game program where the player picks a random number between 1 and 100 that the computer has to guess. (hint: Before you start, think about how you guess and try to program the computer to do the same.)