Tutorial 4:

- Exercises 1 to 5 cover *while loops.*
- Exercises 6 to 11 cover *for loops.*
- Exercises 12 and 13 are extended exercises using *while loops.*

Exercise 1 - Guess the Number

Part A

You translated the following into a flow chart during the lecture.

- Initialise variable *hidden* with a value of 6.
- Ask user to enter a guess (number between 1 and 20).
- Loop while the guess is not the hidden number.
- Within the loop:
    - State guess is not correct,
    - ask user for next guess (number between 1 and 20).
- If the guess is correct the loop will no longer run.
- Let the user know their guess was correct.

Now translate your flowchart into a Python program and test your solution.

Part B

- Amend the previous program so that the hidden number (between 1 and 20) is a random number generated by the program (check last week's tutorial for how to generate a random number).
- Hint: Remember to use *import random* at the top of your code.
- Hint: While testing your program it is useful to print the random number generated so you can test the program is working as specified.

Exercise 2 - Guess the number (Too low/Too high)

Amend the program to let the user know if the guess is too low or too high where appropriate. Use an *if, else* statement.

Exercise 3 - Write a program that will allow a user to enter a number of scores until -9 is entered. When -9 is entered, print the average of the scores entered. Ensure that at least one score has been entered before calculating the average (division by zero would produce an error).

Exercise 4 - Revision of exception handling.

Test the following example with a range of input (float, string, and integer):

```
while True:
    n = input("Please enter an integer: ")
    try:
        n = int(n)
        break
    except ValueError:
        print("Requires a valid integer! Please try again.")
print("You successfully entered an integer.")
```

Exercise 5 - Write a program that allows the user to select from four different menu options on a continuous loop. Control the loop by a Boolean. Choose you own menu options using numbers and use option 4 to quit the program (e.g., select 1 for current account, 2 for savings account, 3 for overdraft or 4 to quit). Include a default for the situation where an unrecognised menu option is input by the user.

Use the following input data to test your solution.

Input	Expected Result	Pass/Fail
1	Displays '1 selected' and program loops	
2	Displays '2 selected' and program loops	
3	Displays '3 selected' and program loops	
4	Displays 'Quit selected' and program ends	
5	Displays 'Option not recognised' and program loops	

For loops (Exercises 6 to 11)

Exercise 6 - Write a program that uses a *for* loop to print the times table for a number entered by the user. Example Output:

Enter the number to print the tables for: 7

7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56 etc…

Exercise 7 - Write a program that uses a *for* loop to read in 5 numbers and output the total.
Exercise 8 - Write a *for* loop that will output all odd numbers between 0 and 50.
Exercise 9 - Write a program which asks the user how many stars are required and then make it display that number of stars across the screen.

Exercise 10 - Write a program that simulates the rolling of two six-sided dice and then counts the number of times that the dice generate a double (e.g. both dice show same value). Roll the dice 100 times and then print the number of doubles. Example output: 'Out of 100 you rolled 20 doubles'.

Exercise 11 - Nested for loops - A loop within a loop.
A) Try the following and check you understand the output of the nested loop code.

```
for number in range(3) :
    print("-----------------------------------------")
```

```
print("Outer loop iteration " + str(number))
# Inner loop
for another_number in range(4):
   print("****************************")
   print("In inner loop iteration " + str(another_number))
```

B) Try the following nested *for loop*.

```
for x in range(1,4):  # print 3 rows
   for y in range(1,4):  # 3 asterisks in each row
      print('*', end='')
   print()
```

C) Then amend the program so that it prints this instead:

```
*
**
***
```

Further While Loop exercises

Exercise 12 - Guess the number (5 Guesses)

Create a new solution for the guess the number game. This solution will only allow a maximum of five guesses.

Part A - Create the **flowchart** for the following

- The program picks a random number between 1 and 20.
- The program allows the user a maximum of five attempts to guess the number.
- For each incorrect guess, the program will let the user know if the correct answer is higher or lower than their guess.
- If the user doesn't guess the number in 5 attempts, end the loop.
- If the user guesses the number within 5 attempts, the program will **break** from the loop.
- When the user has run out of guesses or guessed the correct number display an appropriate message. (Hint: check after the loop to decide which message):
    - let the user know they are correct and how many guesses they took, or
    - let the user know they did not guess the number and what the hidden number was.

Part B – Create the Python program.

Exercise 13 – Extend Exercise 5 (menu options) so that it deals with the situation where an input entered is not a number.

Input	Expected Result	Pass/Fail
b	Displays 'Please select 1,2,3 or 4' and program loops	