Class Exercises - Iteration

Syntax

```
for count in range(1, 10):
        print('Hello')
```

prints Hello 9 times

```
num = int(input('Enter a number in the range 1 to 10 : '))
while num < 1 or num > 10 :
  num = int(input('Enter a number in the range 1 to 10 : '))
```

keeps prompting the user until they enter a number between 1 and 10

Vocabulary

Definite iteration (for loop)
structure used when we know before entering the loop how often we want to repeat.

iteration (while loop)
structure used when we do not know beforehand how often we want to repeat.

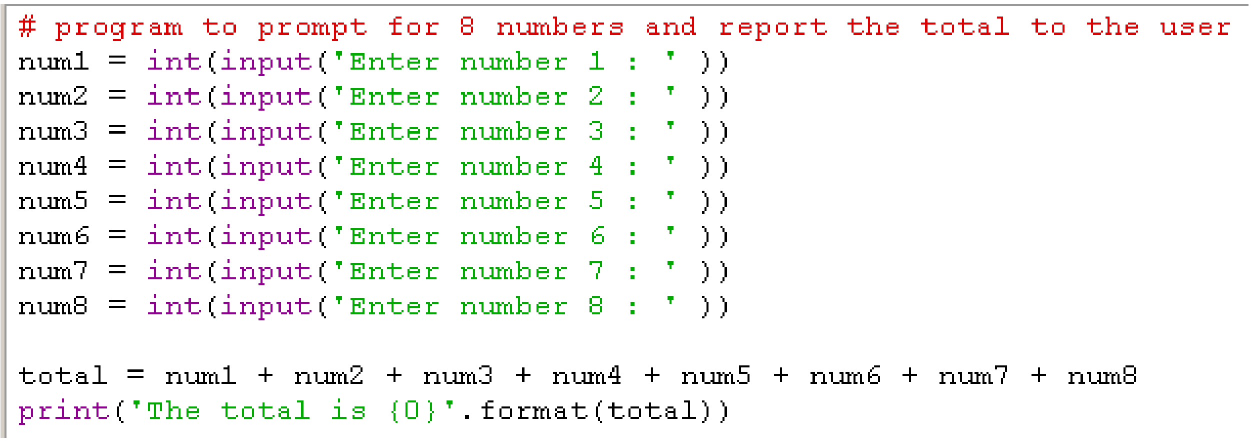
Boolean expression
an expression which checks whether a particular condition is true or false.

Rogue value
a value, which when input, is used to terminate the loop.

Improving a piece of code

Task One

Open the **python** file called **Addition.py** and **refactor** the code so that it uses a more efficient and appropriate style.



```
# program to prompt for 8 numbers and report the total to the user
num1 = int(input('Enter number 1 : ' ))
num2 = int(input('Enter number 2 : ' ))
num3 = int(input('Enter number 3 : ' ))
num4 = int(input('Enter number 4 : ' ))
num5 = int(input('Enter number 5 : ' ))
num6 = int(input('Enter number 6 : ' ))
num7 = int(input('Enter number 7 : ' ))
num8 = int(input('Enter number 8 : ' ))

total = num1 + num2 + num3 + num4 + num5 + num6 + num7 + num8
print('The total is {0}'.format(total))
```

Class Exercises - Iteration

In the next section you may choose the exercises you attempt. There are three types of exercises to consider:

1. Revision Exercises - choose these exercises if you are not confident in your understanding of iteration statements

2. Development Exercises - choose these exercises if you are confident in your understanding but want some more practice

3. Stretch and Challenge Exercises - choose these exercises if you feel you have mastered iteration and want to tackle some tougher problems

Once you feel more confident attempt some of the more difficult exercises. The more practice you get now the more comfortable you will be using iteration in more complex programs later in the course.

Remember, practice makes perfect!

Class Exercises

1. Create pseudocode or flowchart or structure chart first - plan your solution on paper before attempting it
2. Create a set of test data - select values you will enter to test the program and know beforehand what you expect the answers to be
3. Write the program - write the program in Python using the pseudocode to assist you and the test data to ensure the program functions correctly

Revision exercises

Attempt these tasks if you need to build your confidence and understanding of iteration statements.

1. Write a program that will print ‘Hello World’ 10 times.

2. Write a program that will ask the user for a message and the number of times they want that message displayed. Then output the message that number of times.

3. Write a program that asks the user to enter how many numbers are to be averaged, then enter this number of numbers, calculating the average. The program should display the average on the screen.

4. Write a program that asks a user for a number between 10 and 20 inclusive. The program should give the user a message if the number input is outside this range and ask for another number until the number input is within range.

5. Write a program that calculates the average of a series of numbers. Ask the user to enter a series of numbers, entering the rogue value -1 when they have finished. Then calculate the average of those numbers.

6. Write a program that displays a conversion table for pounds to kilograms, ranging from 1 pound to 20 pounds [1 kg = 2.2 pounds]. The values should be shown in neat columns.

Class Exercises - Iteration

Development exercises

Attempt these tasks if you are confident in your understanding but feel you need more practice

7. n factorial usually written n!, is defined to be the product of all the integers in the range 1 to n.
n! = 1 * 2 * 3 * 4 * .... * n. Write a program that calculates n! for a given positive n.

8. Write a program that asks the user to enter the number of stars per row and the number of rows to be displayed. For example, entering 5 and 3 should display:

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

9. Write a program which asks the user for a string and then prints out the string in reverse order. Use a loop to do this.

10. Write a program that finds the largest of a series of numbers. Ask the user to enter a series of numbers, entering the rogue value -1 when they have finished. Then report the largest value to the user.

Stretch and challenge exercises

Attempt these tasks if you feel you have mastered selection and want to tackle tougher problems.

11. Write a program that asks for a number and displays the squares of all the integers between 1 and this number inclusive. It should print 5 values on each line in neat columns.

12. Write a program which will convert a given denary number up to 255 into its 8-bit binary equivalent.

13. Write a program which will convert an 8-bit binary number to its denary equivalent.

14. Write a program which reports the number of words in a phrase typed in by the user. The program will need to count the number of spaces to achieve this. Read the section on string slicing which is about 80% down the page called ‘Additional Exercises 3 – FOR and WHILE LOOPS’ on the Python Summer School.