Week 4: Python Programing - Loops

In this week's session we introduce the concept of repetition, known in programming as *iteration*. This can be done using using either a for loop or a while loop. Getting the hang of these takes practice so will be revisited every week for the rest of the course.

Task 1: In pairs, look at the program below and write out what you think might happen when it runs.

```
from turtle import *

def square():
    for counter in range(4):
        forward(100)
        right(90)

square()
left(45)
square()
```

What would you expect the computer to do? Draw it exactly.

Task 2: Download and run the program and see if it does what you think it might do. You will find it at http://teachinglondoncomputing/ks3-week-4

Did the program run as you predicted? _____

Task 3: Understanding the code

1. Comment the code in the gaps below:

```
def square():
    # write comments after each #
    #
    for counter in range(4):
        #
        #
        forward(100)
        #
        right(90)
        #
```

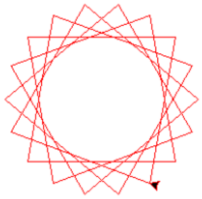
2. Try to spot the errors in the following program (I think there are four - three syntax errors and 1 logic error!)

```
def triangle()
    # this function draws a triangle
    for counter is range[3]:
        forward(120)
        right(90)
```

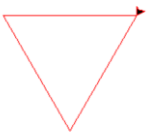
Mark all the errors you can see

3. Look at the following code and decide which picture it will generate. You can type it in to a Python file, then save and run it, to check.

```
def demo():
    pencolor("red")
    for counter in range(20):
        forward(150)
        right(100)
```



PICTURE A

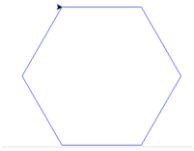


PICTURE B

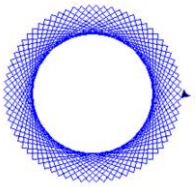
Task 4: Exercises

In pairs, try to modify the starter program to make new programs as follows:

1. Change the starter program to make a blue hexagon using a for loop.

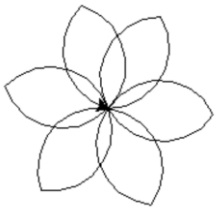


2. Change the angle to 95° and see if you can draw the following shape using a for loop.

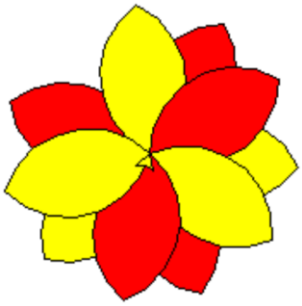


3. The following code will draw a petal of a flower. Use the code in a for loop to make a flower as shown on the right.

```
def petal():
    circle(50,120)
    left(60)
    circle(50,120)
```



4. Now can you make a similar flower in two colours similar to (or better than!) the one below. Use `begin_fill()`, `fillcolor("red")` and `end_fill()` to colour in shapes.



Now EITHER experiment with more Spirograph-style shapes OR try the following exercises using print statements.

5. This code will print "hello" 4 times. Can you write a program to ask the user for their name and print it 5 times.

```
for line in range(4):
    print("hello")
```

6. 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.
7. Write a program that will print the numbers 1 – 10 in order on separate lines, using a `for` loop. Tip: range(10) gives you 0-9 and range(1,11) gives you 1-10.

Extension exercise

Can you write a general purpose function polygon(sides) which will draw a shape of any number of sides? For example polygon(4) will draw a square and polygon(6) will draw a hexagon.

Make a note of any errors that you come across in the notebook and how you fix them.

Key concepts from this sheet:

Term	What it means	How to do it in Python
Iteration	A loop. There are two loops – for loops (a set number of repetitions) and conditional loops or while loops.	`for line in range(4):` `  print("hello")`

Key points to remember about Python (and to remind your students!)

1. The `for` loop is one kind of loop. It uses a counter which takes on a value every time the loop iterates.
2. The `range()` function indicates how many times the statement will repeat. For example `range(4)` means repeat 4 times and range(10) means repeat 10 times. In actual fact, range(4) generates four numbers 0,1,2 and 3 and the counter takes on each value one at a time. Because there are four numbers to take the value of, there are four repeats.