Computer Science	Programming Python Turtle	Year 8	Term 4

Week 1: Python Introduction

A **programming language** is a language that computers understand.

Python is a text-based programming language.

Programming languages are made up of **instructions** that tell a computer what to do.

Commands are little blocks of code that have been pre-written and stored in a **library**. Commands are always followed by brackets.

print is the basic output command used in Python.

IDLE is the development environment that we use. This gives us all the tools that we need to write and test our own code.

The **command shell** will run any code that we type in when we press enter on the keyboard.

An **output** is when signals come out of the computer.

The **print command** outputs the text inside of its brackets onto the screen.

Syntax Errors are when the programming language's rules are not perfectly followed the program will not run.

Logic Errors (i.e. Bug) are when a program runs without any error messages but does not do what is expected of it.

Modules are extra sets of code that we can load into Python in order to add additional functions.

The **turtle module** is an extra set of code that we can load into Python in order to draw.

We **import** the Turtle module by typing in the following instruction:

```
from turtle import *
```

We use the following code to setup our turtle object:

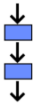
```
turtle = Turtle()
```

The most basic commands are:

```
turtle.right()
turtle.left()
turtle.forward()
```

These commands move the turtle forwards and turn the turtle.

A **sequence** is a set of instructions that the computer carries out one after another.



A **variable** is a named location in memory that can store a value.

The computers need variables to store the *information* we give them and the values they calculate during programs.

In Python to create variables and assign them a value, the equals symbol is used.

```
myNumber = 0
```

All variables are made up of three parts:

1. Name (i.e. Identifier)
2. Type
3. Value

We will focus on the **integer** and **string** data types.

- Strings are alphanumeric. This means they can contain characters, numbers or symbols.

Integers are whole numbers.

An **input** is when signals go into the computer.

The **input command** outputs a message and allows the programs to take an input from the user.

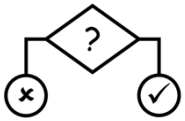
```
input("What is your name?")
```

It is also important that we store whatever the user enters in a variable so we can use it in our program.

```
name = input("What is your name?")
```

To access the value stored in a variable you just use the name of the variable.

```
>>> print(myNumber)
0
```

Week 4: Selection	**Week 5:** Iteration and Subroutines	**Week 6:** Final Project
Selection allows us to decide between different sequences of instruction in a program.  	**Iteration** is when instructions are repeated a certain number of times or until a condition is true. In programming, iteration is often referred to as 'looping'.	Creating an algorithm, such as a list or flow diagram, is a good way to plan to solve this problem.

In Python we use **if**, **elif** and **else statements** for selection.
• If the IF statement's condition is true, the instructions below it are carried out.
• Otherwise if the IF statement's condition is true and the ELIF statement's condition is true, the instructions below the ELIF statement are carried out.
• If all the conditions are false, the instructions below the ELSE statement will be carried out.
 	
`if age >= 70:`
`  print ("You are aged to perfection!")`
 	
`elif age == 50:`
`  print ("Wow you are half a cenury old!")`
 	
`else:`
`  print ("You are a spring chicken!")`
	Iteration allows us to reduce the number of lines in our programs by letting us state that we will repeat certain steps.	Decomposition is when we break down a problem into its simplest form. This makes it easier to solve.

	A subroutine is a self-contained block of code that performs a particular task.
	
	`def drawSquare(sideLength):`
	`  for i in range(4):`
	`  turtle.forward(sideLength)`
	`  turtle.right(90)`
	
	The benefit of subroutines is we can **call** them multiple times to perform a task.
	`drawSquare()`
		When we design subroutines we can add parameters. These are different pieces of data that we would send through when we call the code.
		
		`def drawSquare(sideLength):` for example has ONE parameter – `sideLength`. This is used then inside of the code to move the turtle.

	To lift the pen up and down you can use these commands:
	`turtle.penup()`
	`turtle.pendown()`
	
	To change the colour of the pen you can use these commands:
	`turtle.pencolor("colour")`
	`turtle.pencolor(R,G,B)`
		When we call the subroutine, the specific data that we want to use is sent. This is called the argument.
		
		When we call a subroutine such as `drawSquare`, we then tell it how long we want the sides to be:
		`drawSquare(100)`
		`drawSquare(200)`

	Values can be **passed** to the subroutine for it to use. This is done by putting the **parameters** name in the definition's brackets, and putting the value, called the **argument**, in the call's brackets.	

Extension QR Code – Use the QR code to find out more information on Python turtle.