**Week 2 – Introduction to Python | Day 1**

**Control flows**

Control flows in Python refer to the order in which the statements of a program are executed based on certain conditions or loops. They allow us to dictate the flow of execution in a program, enabling us to make decisions and repeat actions as needed. We have three main control structures; sequential, conditional and loops.

**Day 2**

**Input and Output**

Input and output (I/O) in Python refer to the way a program interacts with the outside world. This includes reading data from users, files, or other sources, as well as displaying information to users or writing data to files.

**Reading and Writing Files**

Reading and writing files in Python is a fundamental aspect of working with data. Python provides built-in functions and methods for performing these operations.

**Read Methods**

In Python, "read methods" typically refer to the methods that are used to read data from a file. These methods are used when you want to access the contents of a file.

**Methods**

Methods are functions that are defined within a class. They are associated with the class and can operate on its attributes (variables) and perform actions related to that class. Methods are a way to encapsulate behaviour within a class.

**Write Methods**

In Python, "write methods" typically refer to the methods that are used to write data to a file. These methods allow you to save information or content to a file.

**My own views on lists and methods**

A list is a collection of items, where each item can be of any data type (such as numbers, strings, or even other lists). Lists are ordered and mutable, which means you can change the elements in a list after it is created. Python provides a variety of methods that can be used with lists to perform operations like adding, removing, searching, and sorting elements.

**File Objects - Reading and Writing to Files**

To write to a file in Python, the open() function is used, with the "w" mode (which stands for write). If the file does not exist, it will be created. If it already exists, it will be overwritten.  
file\_path = "output.txt"  
file = open(file\_path, "w")  
To read from a file in Python, the open () function is used, with the "r" mode (which stands for read). File = open(file\_path, ‘r’)

**DAY 3**

**Data Types**

 Data types are classifications that categorize various types of data based on their nature, characteristics, and the operations that can be performed on them. They define the type of values that a variable can hold, and they determine how those values behave when used in operations.

**Data Types**

 Data types are classifications that categorize various types of data based on their nature, characteristics, and the operations that can be performed on them. They define the type of values that a variable can hold, and they determine how those values behave when used in operations. We have integers, Booleans, floating point and many other data types in python.

**Integers**

 Integers (int) are a data type used to represent whole numbers, both positive and negative, without any decimal or fractional component. They can be of any length and are limited only by the memory available on a machine.

**Floating point numbers**

 Floating-point numbers in are a data type used to represent real numbers, which can have both an integer part and a fractional part. They are called "floating-point" because the decimal point can "float"; that is, it can support a variable number of digits before and after it.

**Unpacking Argument Lists**

 A string is a data type used to represent text. It is a sequence of characters enclosed in either single quotes (') or double quotes ("). Python doesn't distinguish between characters and strings of length one; they are both treated as strings.

**Lambda Expressions**

 A lambda expression in Python is a small, anonymous function that can have any number of parameters, but it can only have one expression. Lambda functions are also known as "anonymous functions" because they don't have a name like a regular function defined with the def keyword.  
The syntax is:  
lambda arguments: expression

**Conventions about the content and formatting of documentation strings**

 Documentation strings, often referred to as docstrings, are used to provide information about the purpose and usage of functions, classes, modules, and packages in Python. They help improve code readability and serve as a form of inline documentation.

**Activity 1**

 x = bool()  
y = bool()  
print ('Enter x as 1 or 0:')  
x = int(input())  
print ('Enter y as 1 or 0:')  
y = int(input())  
z = str(not bool(x or y))  
print ('The Boolean value of x is', str(bool(x)))  
print ('The Boolean value of y is', str(bool(y)))  
print ('The Boolean value of (x or y) is', str(bool(x or y)))  
print ('The Boolean value of (x not y) is', z)

**Day 3 Reflections**

 Understanding and using the appropriate data types is crucial for writing efficient and correct code. It helps ensure that the program handles data in the way it's intended to, preventing unexpected behavior or errors. Additionally, it helps manage memory allocation and utilization efficiently.

**Activity 2**

 def print\_animal\_info(animalType, animalSpecies, waterConsumption):  
print(f"The {animalSpecies} is a type of {animalType}.")  
print(f"It drinks {waterConsumption:.2f} litres of water per day.")  
# Taking user input for animal type, species, and water consumption  
animalType = input("Enter the type of animal: ")  
animalSpecies = input("Enter the species of the animal: ")  
waterConsumption = float(input("Enter how many litres of water the animal drinks a day: "))  
# Calling the function to print the information  
print\_animal\_info(animalType, animalSpecies, waterConsumption)

**Day 5**

 Python syntax refers to the rules and structure that govern how Python code is written and interpreted by the Python interpreter. It defines how programs written in the Python language should be formatted and organized to be considered valid and executable.