**Experiment # 2: Introduction to Python - II**

**Objective:** To provide basic knowledge about the python language; Tuples, Sets, Dictionaries, Conditional and Loop statements

**Time Required** : 3 hrs

**Programming Language** : Python

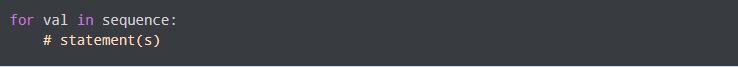
**Software Required** : Anaconda/ Google Colab

**Python — For Loop**

A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string). This is less like the for keyword in other programming languages and works more like an iterator method as found in other object-orientated programming languages.

With the for loop, we can execute a set of statements, once for each item in a list, tuple, set etc.

In Python, the for loop is used to run a block of code for a certain number of times. The for loop does not require an indexing variable to set beforehand



Here, *val* accesses each item of sequence on each iteration. Loop continues until we reach the last item in the sequence.

You can loop through the tuple items by using a for loop.

A screenshot of a computer

Description automatically generated

Shape

Description automatically generated with medium confidence

**Python —While Loop**

Python while loop is used to run a block code until a certain condition is met.

The syntax of while loop is:



**Example:**

Text

Description automatically generated

**Python Tuples**

Tuples are used to store multiple items in a single variable. Tuple is one of 4 built-in data types in Python used to store collections of data, the other 3 are List, String, and Dictionary, all with different qualities and usage.

A tuple is a collection which is ordered and unchangeable. It allows duplicate members. Tuples are written with round brackets.

A screenshot of a computer

Description automatically generated

**Python - Access Tuple Items**

You can access tuple items by referring to the index number, inside square brackets:

A screenshot of a computer

Description automatically generated

**Negative Indexing**

Negative indexing means start from the end. -1 refers to the last item, -2 refers to the second last item etc.

A screenshot of a computer

Description automatically generated

**Python - Update Tuples**

Tuples are unchangeable, meaning that you cannot change, add, or remove items once the tuple is created. But there are some workarounds.

*Change Tuple Values*

You can convert the tuple into a list, change the list, and convert the list back into a tuple.

A screenshot of a computer

Description automatically generated

**Python - Tuple Methods**

Python has two built-in methods that you can use on tuples.

A screenshot of a computer

Description automatically generated

**Python Sets**

A set is a collection which is unordered and unindexed. It does not allow duplicate members. In Python, sets are written with curly brackets.

Graphical user interface, text, application

Description automatically generated

**Access Items**

You cannot access items in a set by referring to an index or a key but you can loop through the set items using a for loop, or ask if a specified value is present in a set, by using the in keyword.

Text

Description automatically generated

**Add Items**

To add one item to a set use the add() method.

Graphical user interface, text

Description automatically generated

To add more than one item to a set use the update() method.

Text

Description automatically generated with medium confidence

**Get the Length of a Set**

To determine how many items a set has, use the len() method.

Graphical user interface, text, application

Description automatically generated

**Set Methods:**

**Table

Description automatically generated**

**Python Dictionaries**

A dictionary is a collection which is unordered, changeable, indexed and doesn’t allow duplicates. In Python dictionaries are written with curly brackets, and they have keys and values.

Graphical user interface, text, application

Description automatically generated

**Accessing Items**

You can access the items of a dictionary by referring to its key name, inside square brackets:



There is also a method called get() that will give you the same result:



**Loop Through a Dictionary**

You can loop through a dictionary by using a for loop. When looping through a dictionary, the return value are the keys of the dictionary, but there are methods to return the values as well.

Text

Description automatically generated

**Example:**

Text

Description automatically generated

**Dictionary Length**

To determine how many items (key-value pairs) a dictionary has, use the len() function.



**Dictionary Methods**

A screenshot of a computer

Description automatically generated

**Python Conditions and If statements**

Python supports the usual logical conditions from mathematics:

* Equals: a == b
* Not Equals: a != b
* Less than: a < b
* Less than or equal to: a <= b
* Greater than: a > b
* Greater than or equal to: a >= b

These conditions can be used in several ways, most commonly in "if statements" and loops.

An "if statement" is written by using the if keyword.

If statement:

Text

Description automatically generated with medium confidence

In this example we use two variables, a and b, which are used as part of the if statement to test whether b is greater than a. As a is 33, and b is 200, we know that 200 is greater than 33, and so we print to screen that "b is greater than a".

**Indentation**

Python relies on indentation (whitespace at the beginning of a line) to define scope in the code. Other programming languages often use curly brackets for this purpose.

*If statement, without indentation (will raise an error):*

Graphical user interface, text, application, chat or text message

Description automatically generated

**Elif**

The elif keyword is python’s way of saying "if the previous conditions were not true, then try this condition".

Graphical user interface, text

Description automatically generated  
In this example a is equal to b, so the first condition is not true, but the elif condition is true, so we print to screen that "a and b are equal.

**Else**

The else keyword catches anything which isn't caught by the preceding conditions.

Text

Description automatically generated

In this example a is greater than b, so the first condition is not true, also the elif condition is not true, so we go to the else condition and print to screen that "a is greater than b". You can also have an else without the elif.

Graphical user interface, text

Description automatically generated

**Short Hand If**

If you have only one statement to execute, you can put it on the same line as the if statement.

*One line if statement:*

Graphical user interface, text

Description automatically generated with medium confidence

**Short Hand If ... Else**

If you have only one statement to execute, one for if, and one for else, you can put it all on the same line:

*One line if else statement:*

A picture containing text

Description automatically generated

This technique is known as **Ternary Operators**, or **Conditional Expressions**. You can also have multiple else statements on the same line:

*One line if else statement, with 3 conditions:*

A picture containing text

Description automatically generated

**Logical Operator**

**And**

The ***and***keyword is a logical operator, and is used to combine conditional statements.

*Test if a is greater than b, AND if c is greater than a:*

Graphical user interface

Description automatically generated

**Or**

The ***or*** keyword is a logical operator, and is used to combine conditional statements.

*Test if a is greater than b, OR if a is greater than c:*

Graphical user interface, text

Description automatically generated

**Nested If**

You can have if statements inside if statements, this is called *nested* if statements.

Graphical user interface, text

Description automatically generated

The while loop requires relevant variables to be ready, in this example we need to define an indexing variable, i, which we set to 1.

**The break Statement**

With the break statement we can stop the loop even if the while condition is true.

*Exit the loop when i is 3:*

**Text

Description automatically generated with low confidence**

.

**Looping Through a String**

Even strings are iterable objects, they contain a sequence of characters.

*Loop through the letters in the word "banana":*

Graphical user interface, text, application

Description automatically generated

**The break Statement**

With the break statement we can stop the loop before it has looped through all the items.

*Exit the loop when x is "banana":*

Graphical user interface, text, application

Description automatically generated

**The range() Function**

To loop through a set of code a specified number of times, we can use the range() function. The range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and ends at a specified number.

Graphical user interface, text

Description automatically generated

**Note:** The range(6) is not the values of 0 to 6, but the values 0 to 5.

The range() function defaults to 0 as a starting value, however it is possible to specify the starting value by adding a parameter: range(2, 6), which means values from 2 to 6 (but not including 6).

*Using the start parameter:*

Graphical user interface, text, application

Description automatically generated

The range() function defaults to increment the sequence by 1, however it is possible to specify the increment value by adding a third parameter: range(2, 30, **3**).

*Increment the sequence with 3 (default is 1):*

Text

Description automatically generated with medium confidence

**Else in For Loop**

The else keyword in a for loop specifies a block of code to be executed when the loop is finished.

*Print all numbers from 0 to 5, and print a message when the loop has ended:*

Text

Description automatically generated

**Note:** The else block will NOT be executed if the loop is stopped by a break statement.

*Break the loop when x is 3, and see what happens with the else block:*

Text

Description automatically generated

**Nested Loops**

A nested loop is a loop inside a loop. The "inner loop" will be executed one time for each iteration of the "outer loop".

*Print each adjective for every fruit:*

Text

Description automatically generated

**The pass Statement**

for loops cannot be empty, but if you for some reason have a for loop with no content, put in the pass statement to avoid getting an error.



**LAB TASKS:**

**Q1:** Write a program that takes two integers as input (lower limit and upper limit) and displays all the prime numbers including and between these two numbers.

**Q2:** Given a list iterate it and display numbers which are divisible by 5 and if you find number greater than 150 stop the loop iteration.

list1 = [12, 15, 32, 42, 55, 75, 122, 132, 150, 180, 200]

**Q3:** Write a program that accepts a comma separated sequence of words as input and prints the words in a comma-separated sequence after sorting them alphabetically. Suppose the following input is supplied to the program: *without, hello, bag, world*. Then, the output should be: *bag, hello, without, world.*

**Q4:**

1. Write a simple calculator program. Follow the steps below:

* Declare and define a function named Menu which displays a list of choices for the user such as addition, subtraction, multiplication, and classic division. It should take the choice from user as an input and return.
* Define and declare a separate function for each choice (each mathematical operation).
* In the main body of the program call the respective function depending on the user’s choice.

1. Implement the following functions for the calculator you created in the above task.

* Factorial
* x\_power\_y (x raised to the power y)

**References:**

<https://www.w3schools.com/python/default.asp>