Week - 1: Python Syntaxes

Introduction:-

You will get familiarise in working with the basic syntax of Python and introduced to certain programming basics off Python language.

The first module includes four sessions:

1st session(Week 1):-

- **Basics of Python**: This session introduces you to Python Programming and the environment you require for coding. Once you have the basic setup ready, you will learn to write your first program in Python, followed by learning about different data types. Finally, towards the end of the session, you will look at various arithmetic and string operations supported by Python.

2nd session(Week 2):-

- **Data Structures in Python**: The session starts with Introducing various data structures in Python, which include tuples, lists, sets and dictionary. Further, you learn about all these data structures in detail and various operations related to them.

3rd session(Week 3):-

- **Control Structure and Functions in Python**: This session is the essence of programming since they help computers do what they do best—automate repetitive tasks intelligently. It incorporates all the decision making control structures and functions supported by Python.
- **OOP in Python**: This session will teach you about the various object-oriented programming methodologies in Python that includes classes, objects and methods.

Getting Started with Python:-

Python is a general-purpose programming language that was named after **Monty Python(i.e.** were a British surreal comedy troupe**)**. It is simple and incredibly readable since it closely resembles the English language. But still, why should you use Python?

Python is a language that finds use in nearly every domain possible. Its official **website** will give you an overview of this. In addition, its simplicity, as well as the way it ensures tasks can be performed using fewer lines of code, is encouraging many developers across the world to take it up.

Little about **Anaconda:-**

Anaconda is an open-source distribution that simplifies package management

and deployment. The package management system 'Conda' manages package versions.

Advantages of using Anaconda

1. It is easy to manage and supports most of the libraries required for Machine learning/Artificial Intelligence problems.
2. Anaconda comes with many libraries such as NumPy, OpenCV, SciPy, PyQt, the Spyder IDE, etc.

Little about **Jupyter Notebook:-**

You will use the Jupyter IPython Notebook as the main environment for writing Python code throughout this program. The key advantage of using Jupyter Notebook is that you can write both code and normal text (using the Markdown format in Jupyter) in the notebooks. These notebooks are easy to read and share, and can even be used to present your work to others. Here is a brief overview of Jupyter Notebook.

Introduction to Jupyter Notebook:-

Welcome to the Jupyter Notebook introductory session. You will use the Jupyter IPython Notebook as the main environment for writing Python code throughout this program and because of this reason, learning how to use the various functionalities present in the Jupyter Notebook is extremely crucial so that your coding experience going forward can be smooth. The main advantage of using Jupyter Notebook is that you can write both code and normal text (using the Markdown format in Jupyter) in the Notebooks. These notebooks are easy to read and share, and can even be used to present your work to others.

Here's a brief of the concepts in the notebook.

Headings

- # for the titles
- ## for the main headings
- ### for the subheadings
- #### for the smaller subheadings
- ##### for the italic subheadings

Emphasis

- __string__ or **string** for bold text
- _string_ or *string* for italic text

Line breaks

-
 wherever you want a line break, as the notebook sometimes doesn`t give you the required line break where you want it

Indenting

- > to indent the text
- >> for further indenting it, and so on

Bullets and numbering

- A single dash, i.e. - followed by two spaces to make bullet points
- A number and a dot followed by a space, i.e. 1. to make numbered lists

Next, it is also crucial that you know about the various shortcuts while using the Jupyter Notebook.

Command mode shortcuts

- Esc: To go into command mode
- Enter: To go back to edit mode
- M: To convert a cell to a markdown cell
- Y: To convert a cell back to a code cell
- A: To insert a new cell above
- B: To insert a new cell below
- D + D: To delete cell
- Z: Undo the last operation
- F: To find and replace on your code
- Shift + Up/Down: To select multiple cells
- Space: Scroll notebook downwards
- Shift + Space: Scroll notebook upwards

Edit mode shortcuts

- Shift + Enter: To execute the code in the current cell and go to the next cell
- Alt + Enter: To execute the code in the current cell and insert new cell below
- Shift + Tab: To get a brief documentation of the object that you have just typed in the coding cell
- Ctrl + Shift + -: To split the cell at the cursor
- Shift + M: To merge selected cells

We have also provided a link to all the shortcuts for Mac users below.

- Jupyter Notebook Mac shortcuts

You will slowly get used to effortlessly using these commands to write codes efficiently on your Jupyter Notebook as you move forward, so do not worry about memorising these commands all at once right now.

Data Types in Python:-

You have learnt that variables are nothing but a memory location to store values assigned to them. Some of the properties of these variables are:

1. There is no need to declare the data type of the variable as done in other programming languages, such as C, C++ and JAVA

int c = **5**

string name = 'Alex'

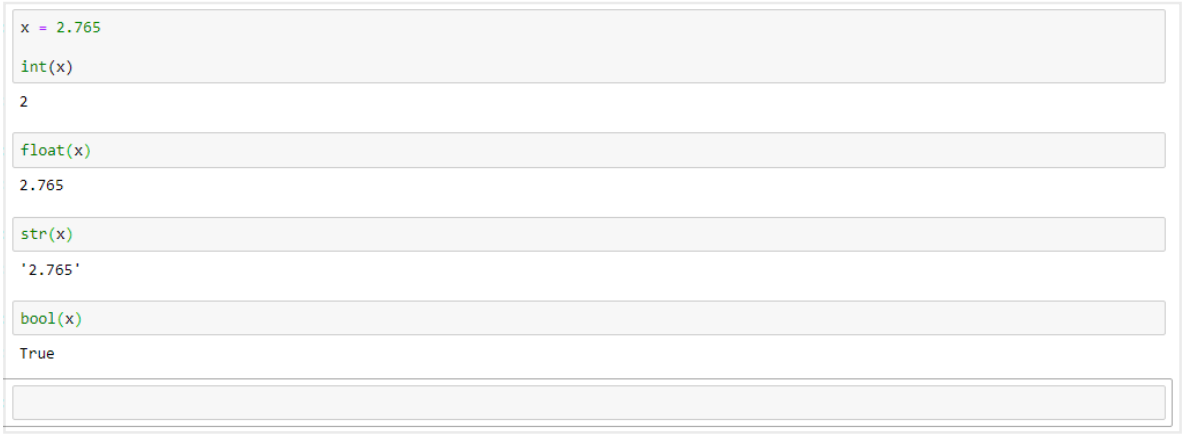
2. The variable name (or identifier) cannot start with a number, i.e., declaring something like 2name = 7 throws an error.
3. Python is case sensitive or in other words, these variables are case sensitive. This means that declaring name= 2 & Name = 4 would create two different variables.

In the previous video, you understood how to declare a variable in python. Let's use the variables to make a small application that can calculate the age of a person in months.

In the video, you learnt two commands:

- To find the data type of a variable use type()
- For casting variables use the keyword of the target data type.

In order to change the data type of a particular variable, you can simply call the following inbuilt methods in python.



In the above snapshot of code, you are assigning a value to a variable (x) and then using typecasting, converting it into different data types. So for example, when you convert x into an integer it converts the floating-point value into an integer value.

In the video, you learnt two commands:

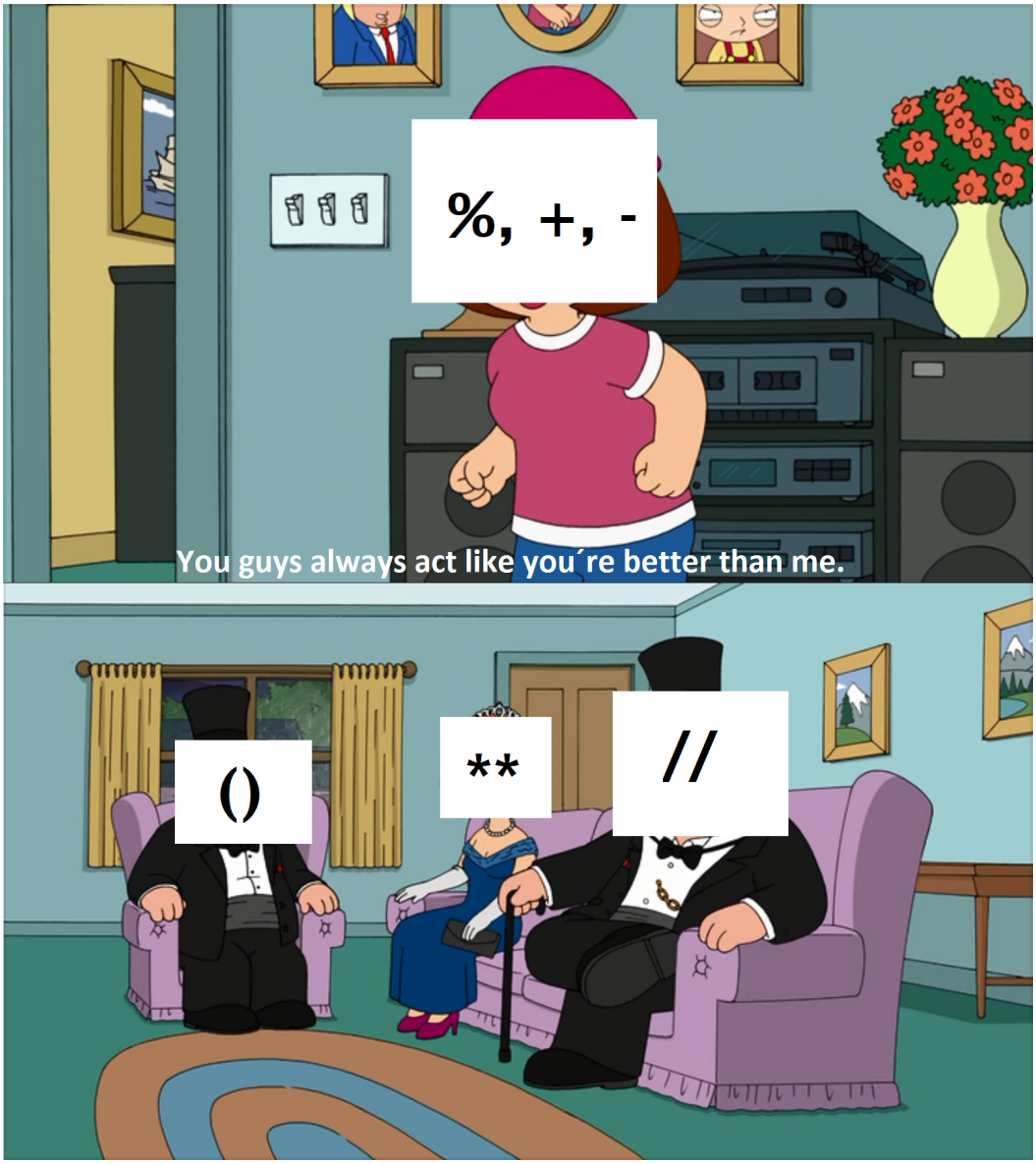
To find the data type of a variable use type()
For casting variables use the keyword of the target data type.
In order to change the data type of a particular variable, you can simply call the following inbuilt methods in python.

Arithmetic Operations:-

In the previous segment, you learned about various data types in Python, next let us learn about Arithmetic operations. Arithmetic operations are an integral part of every programming language. Let us understand how you perform them in Python from the following video:

To do some mathematical operation which involves one or more than one operator, we did follow some rules. The same is the case in Python, where if multiple operations have to be done in a single problem, then we make use of operator precedence rule.
Remember the Rule PEDMAS-which is acronym for Parentheses, Exponents, Division, Multiplication, Addition and Subtraction.

A meme to help you remember the operator precedence rule. :)



String Operations:-

90% of the data in the world is in form of text data; as we have seen in the last few segments, python handles text data generally via the string data type; and this makes '**strings**' perhaps one of the most important data types and hence, learning how to work with strings is crucial.

you understood the basics of the string data type, and you also understood the functionalities a string could provide, such as the escape character or '\' operator.

The application that we built in the above video has used string concatenation to order a dessert. The code used in the video is given below.

#Let's try building a system that could ask for flavour, and the type of desert user wants

```
flavour=input("What flavour would you like ")
dessert_type=input("What type of dessert would you like ")
print("You have ordered", flavour+"-"+dessert_type)
```

String concatenation is just one of the way to manipulate a string, let us have a look at what other string manipulation techniques does Python offer.

In the previous video, you learned about indexing in strings. Always remember that forward indexing starts with **0,** and reverse indexing starts with **-1.**
We should also keep in mind the distinct feature of immutability that string data type provides, which means that once a string is created, it cannot be changed.

And for cases where we may want to add data from multiple strings together, such as creation of a variable 'Full name' from variables 'First name' and 'Last name' we'll use string concatenation.

Earlier in the segment, you saw how to do indexing and slicing in strings. Now let us take a step forward and learn how to change the character-case for a string or remove unwanted characters adjacent to a given string variable.

Apart from the methods you saw in the video, there is one more important method which you might want to use in certain situations. The split() method splits the string into substrings based on the separator chosen; the final outcome is a list which has the substrings in it.

```
a = "Hello World!"
print(a.split(" ")) # returns ['Hello', 'World!']
```

This method is used in situations where you might want to separate certain strings. For example, consider the categories of products in an e-commerce data set. It might be possible that they are given in the following manner:

- electronics-phone
- electronics-headset
- furniture-table

Now with the data structured in this manner in order to get a category and the sub-category level data, we'll have to split the words into two sub-strings.

Summary:-

First, we started exploring the very basics of Python and then moved on and wrote our first Python program. Further which we saw the different data types supported by Python and the various arithmetic and string operations; finally, we attempted the coding questions based on a practice exercise.

In the next session, you will have a look at data structures that could store multiple values together.

Data structures are typically data containers that could store numerous values together-your driving license details, which might include numeric, text, or alphanumeric values, can be stored in a single data structure.
Up next, we will dive deep into various data structures supported by Python and learn about each one of them in detail.