Python for Beginners: Part I

*This will be a three-part series introducing Python for Data Science.*

*\*Please note, we will be using Python 3.0 and above.*

Introduction:

Python as a programming language was conceived in the early 1980s, but its implementation began much later in 1989. Its founder, Guido van Russom, describes Python as a: “high-level programming language and it’s core design philosophy is all about code readability and a syntax which allows programmers to express concepts in few lines of code”.

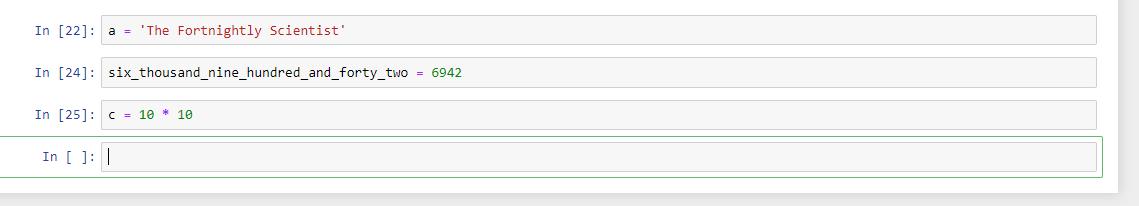
Nowadays, Python lives up to these words. It has seen a huge spike in usage of late, particularly because of its ease of use, coupled with its immense usefulness in Data Science, emerging as the best language to learn in 2019!

The Basics:

First and foremost you will need a coding ‘environment’. I personally use Jupyter Notebooks, which we will be using for the projects in the Deep Learning series

Variables

Variables are the building blocks of Python. In Python, they are similar to the ones we use in algebra. They can be used to represent values, perform operations, and can even be modified to . Let’s look at a couple of examples:



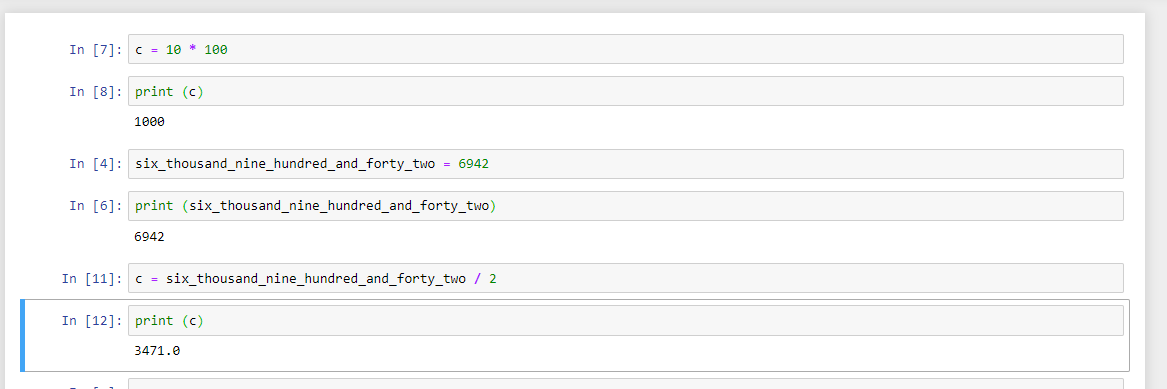
In the example above, *‘a’* is assigned the value of The Fortnightly Scientist. When assigning a string to a variable, always put it inside apostrophes (‘’). The variable *six\_thousand\_nine\_hundred\_and\_forty\_two* has the integer value of ‘6942’. Variables can also be operations of two numbers (or even two variables). In this case, the value of *‘c’* will be 100. The name of a variable can be anything except a few reserved **keywords**.

***Tip*** *- We should always strive to give our variables sensible names, thereby helping us to easily identify them.*

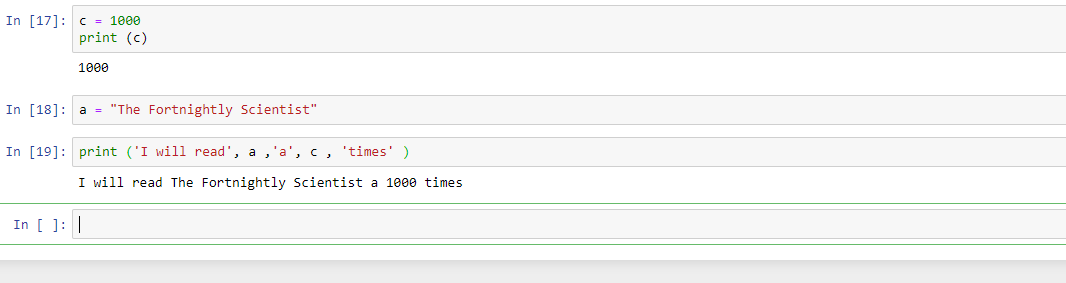
But how do we check if a variable has the value we assigned? The ‘print’ statement comes to our rescue!

Print():

The **print** function in Python helps us to visualize the values by ‘printing’ them. In Python 3.0, the Print function is always called with a parenthesis(), with our variable(s) going inside it.



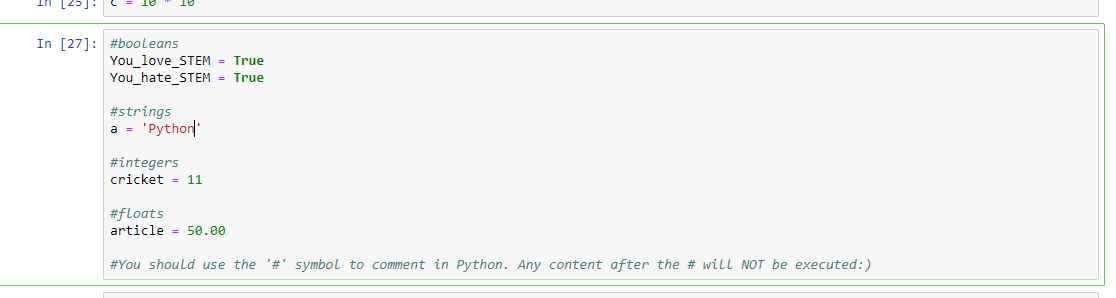
In our example, both the variables have printed out the values, as expected. It is possible to give one variable the value of another. Additionally, we can modify this variable while assigning it to a new one. In this case, the value of ‘*c*’ was 100. However, we then re-assigned it to be the value of the variable *‘six\_thousand\_nine\_hundred\_and\_forty\_two*’ dividedby 2. Therefore, we reached the value of 3471. Additional strings can also be added, while printing, by using the apostrophe (‘) for denoting non-variables and the comma (,) to separate each part.



As seen above, we have used two variables as well as some strings to get this sentence.

*Activity:Modify the above code such that instead of 1000 times, the value should* ***always*** *be twice of what ‘c’ is.*

Here are some other data types:

**

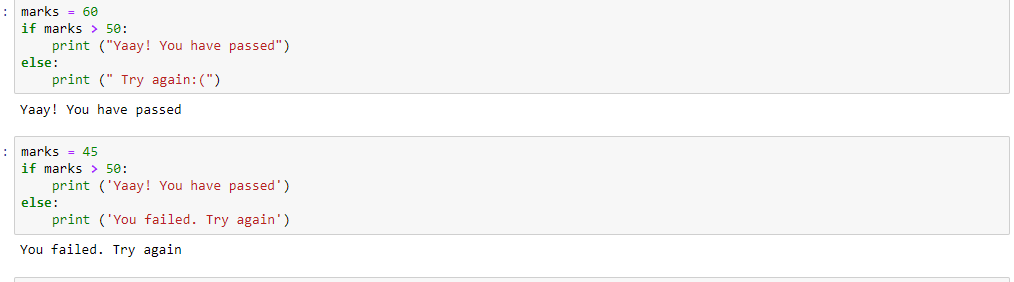
***Note*** *- Use the ‘****#****’ symbol to comment out anything in Python. The computer will skip reading and executing any line starting with ‘****#****’. Comments are only for other coders and yourself in the future to elucidate your intentions when creating that piece of code.*

Conditional Flow:

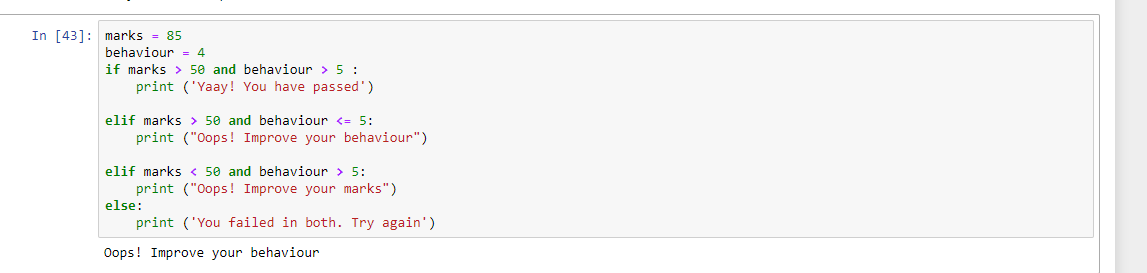
Conditional flow in Python is used to execute commands based on some condition(s).

If/Else/Elif statements

The **if** and **else** statements are used in Python used to make conditional statements. The **if** statement always comes before the **else** statement. The **if** statement states the condition, for which the action specified in the **else** statement will be executed. Here is an example illustrating this concept:



Here, by changing the number of marks, we can alter the result, i.e. the printed statement. Note that we always write the commands under an **if** or **else** statement, with a slight indentation, i.e. we leave some space at the beginning. Hence, this is called indentation.



In the above case, we have used a keyword **‘elif’**. This can be used when there are more than two scenarios. The **‘and’** keyword is used when two or more conditions have to be fulfilled in order for the command to execute. Similarly, the keyword **‘or’** can be used when only one of the conditions have to be executed in order for the command to be executed.

*Activity:*

*How would you modify the above code such that those whose marks are higher than 80 can pass, even if they have a behaviour score of only 2 or more?*

**Challenge:**

You are tasked with building a system that automatically accepts or rejects applicants into a college. The criteria for acceptance/rejection are as follows:

* SAT score is the most important criteria. It is scored out of a 100, and all scores above 90 are accepted **regardless** of other aspects.
* Extarciruculars are the next most important criteria. Extracurriculars are scored out of 10. A student with an SAT score between 50 - 90, can get in with an Extracirucular score of greater than or equal to 5.
* A student who has not scored more than 90 in SAT and has got a Extracurricular circular score less than 5, can get in only if he has been a part of TFS.
* A student filling neither of the above criterias will be rejected, and asked to apply again.

In summary:

* Variables
* Print Statement
* Conditions
* If/else/elif

In the next installment of this series we will be learning all about ‘for’ and ‘while’ loops, as well as functions in Python.

To receive notification of part two in this series, please subscribe below, and follow us on social media at: