Programming in Python for Business Analytics (BMAN73701)

Week 1

Lecture 1

Dr. Fanlin Meng ([fanlin.meng@manchester.ac.uk](mailto:fanlin.meng@manchester.ac.uk))

Office - AMBS 3.094

Virtual office – <https://zoom.us/j/94473763354>

side note: any line written between # is representing a line of code that can be used

**Week 1**

**Lecture 1 (course introduction + setting up your analysis environment)**

\*\* Slides 18-20

Setting up our analysis environment

Installing Anaconda and Spyder

* Anaconda – Freemium open-source software for large scale data processing, predictive analytics, and scientific computing, that aims to simplify package management and deployment.
* Spyder – Open-Source cross-platform IDLE (Integrated DeveLopment Environment) for scientific programming in Python language.

\*\* Slide 21

If we are in a python interpreter, we can start writing whenever there is the sign “>>>”. The commands can be written one-at-a-time in the shell or interpreter and results can be seen immediately by hitting enter (return for mac).

\*\* Slides 22-25

A black background with green text

Description automatically generatedYOUR FIRST PROGRAM

A black screen with green text

Description automatically generated

A screenshot of a computer

Description automatically generated\*\* Slides 26-29

USING PYTHON AS A CALCULATOR

* Slides 26-27
* A screenshot of a computer

  Description automatically generatedUseful operators:
  + + : Addition
  + - : Subtraction
  + / : Normal Division
  + () : Parenthesis (can be used for grouping)
  + // : Floor Division (converts a floating

quotient to an integer be removing.

the decimal places)

* + A screenshot of a cell phone

    Description automatically generated% : Remainder
  + \*\* : Powers
  + = : Assign Values to a variable
  + \_ : Last printed expression is assigned

to the variable “\_” (underscore)5

* Slide 29
* A black background with green text

  Description automatically generatedWe can import ‘math module’ which can help us do complicate and sophisticated calculations.

\*\* Slide 30

BUILT-IN NUMERIC DATA TYPES

* int : Integer numbers (1, 2, 3, -4, -5, -9, etc.)
* float : Decimal numbers (5.0, 4.20, 6.9, etc.)
* complex: Complex numbers; use j or J to indicate the imaginary part (3+5j, 9+8J, etc.)

A black background with green letters and a plus

Description automatically generated

\*\* Slide 33-34

A black background with green text

Description automatically generatedSTRINGS

* If want to use text, use a string
* Strings can be used by using ‘ ’ or “ ”.
* A green text on a black background

  Description automatically generatedIf we are using print() command, it is going to be displayed as normal text, but in general, a string is displayed using single quotes, ‘ ’.
* If we want to include some characters like double quotes, we

can’t do that directly, so we use blackslash (\) to use them.

* A table for these characters is:

A screenshot of a computer

Description automatically generated

\*\* Slide 35

OUTPUT: THE print() FUNCTION

* A black background with green text

  Description automatically generatedThe print function gives a more readable output, by omitting the enclosed quotes and also printing escaped and special characters.

\*\* Slide 36

INPUT: THE input() FUNCTION

* This prompts the user to input data.
* It returns it if we are working in a terminal.
* We can also assign a variable to this input and store the information given by the user.

A black and white text

Description automatically generated with medium confidence

**A black background with green text

Description automatically generated**\*\* Slide 37-39

STRING OPERATIONS: **CONCATENATION**

* Does not matter If your strings are made/combined with ‘ or “.
* Numbers are concatenated as strings ONLY.
* Strings can be multiplied by integers (BUT NOT by other

A black background with green text

Description automatically generatedstrings or floats).

* Within the print command, concatenation can be done 2 ways:
  + Using +
  + Using ,

A computer screen with green text

Description automatically generatedUsing , is more convenient as it considers each item to be concatenated independently from each other.

\*\* Slide 42

QUIZ

* Question: Write a small program (using the editor) that asks the user for her year of birth, and then output “You will be x years old in 2020”, where x is the actual age of the user in 2020.

A computer screen with a green arrow

Description automatically generated with medium confidence

\*\* Slide 43

TYPE CONVERSION

* If we have some numbers in text form and want to multiply it, we can’t as we can only add strings, so in this case we can use type conversion such as int(), float(), str(), which will convert the input in parenthesis to the desired type.

A black background with green text

Description automatically generatedA screen shot of a computer

Description automatically generated

\*\* Slides 45-47

VARIABLES

* A variable allows us to store any information that we want to and store any input that we take from the user. (can be used by operator ‘=’)
* Variables can be reassigned arbitrarily (means if we assign any value to a variable in which we had already stored something, it WILL BE overwritten without any warning).
* They are of dynamic semantics (means any kind of type can be stored in a variable).
* A screen shot of a computer code

  Description automatically generatedVariables name can ONLY consist of “**letters**, **numbers** and **underscores(\_)**, and they CAN’T start with numbers.
* if a variable is not used and then called, it gives an ERROR.
* We can use “del” command to DELETE a variable, which means the

value which was stored in it.

A black background with green text

Description automatically generated

\*\* Slides 49-50

IN-PLACE OPERATORS’

* In-Place operators helps us to write arithmetic operations such as “x = x + 7”, more concisely as

A black background with green symbols

Description automatically generatedA black background with green text

Description automatically generated“x += 7”.

A screenshot of a black background with green text

Description automatically generated

A black background with green symbols

Description automatically generated

Revision before lecture 2

* Numeric Data types: **int, float, complex**
* Text Data types: **string**

**Lecture 2 (Conditionals and Loops)**

* Summary
  + Sequence types (list, tuple, range, etc.)
  + Mapping types (dictionary, etc,)
  + Boolean data type (basically means TRUE and FALSE)
  + None Type: type for the none object, indicating no value (nothingness, nothing, NULL) (this is not 0, different from that because 0 in itself is a value but here it is ‘nothing’)

\*\* Slide 6-7

COMMENTING YOUR CODE PROPERLY

* # or triple double quotes (“ “ “) are used for commenting properly on your code.
* Commenting which means leaving comments in your code ensures not only we understand our code but also if we give our code to someone else, they can understand it easily with the help of comments.
* This also makes code more readable and it’s also going to be considered in coursework assessment.

A screenshot of a computer code

Description automatically generated\*\* Slide 8

\*\* Slide 10

BUILT-IN TYPES

Principal built-in types are **Numerics** (did in the last lecture, i.e., int, float, complex)**, Sequences, mappings, classes, instances and exceptions.**

* basic sequence types are **list, tuple, range**

\*\* Slide 11

SEQUENCE TYPES

* **LIST**
  + most versatile
  + typically used to store homogenous items (same type) but it can be used for different types
  + Initialization of a list
    - if a list if of n items: # list\_name = [item\_0, item\_1, item\_2, … , item\_n-1]
    - for an empty list we can use: # list\_name = list() # or # list\_name = [] #
  + they are mutable, meaning they can be edited after their creation
    - operations such appending, removing, changing can be done after assignment
  + **For accessing items in a list**
    - we can use “indexing” and “slicing”
    - Indexing is for the purpose to extract a single element/item
    - Slicing is more useful as it enables the user to extract a SET OF ELEMENTS.
    - A green text on a black background

      Description automatically generatedA screenshot of a computer

      Description automatically generated