COMP809 – Data Mining and Machine Learning

Lab 1 – Python Basics – Task 1

Objective of this lab is to give students an understanding about Python environment and basics of Python programming.

Introduction

1. Python Version

To check the Python version installed on your computer open the command line and type Python:

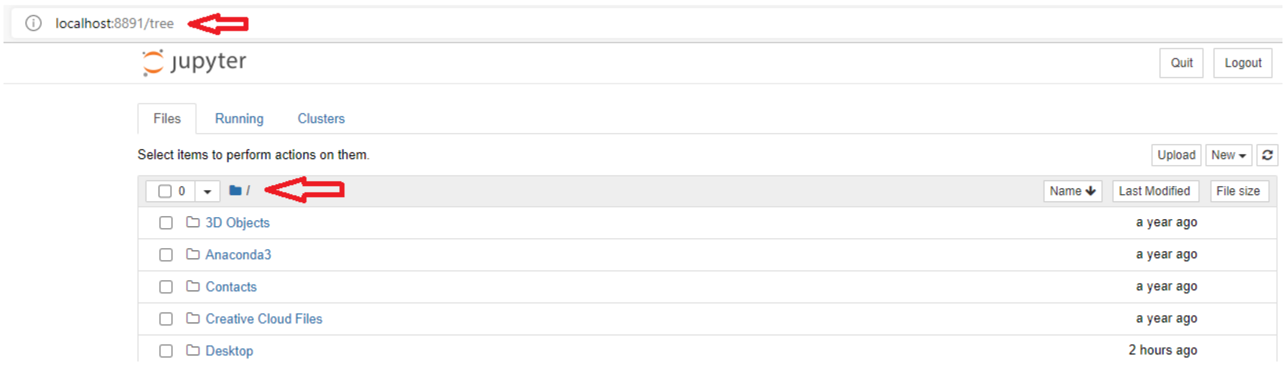
```
C:\Users\szandi>Python
Python 3.10.2 (tags/v3.10.2:a58ebcc, Jan 17 2022, 14:12:15) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

2. Getting started with Jupyter

Various Python IDE (Integrated Development Environment) and code editors such as PyCharm, Spyder, Sublime Text etc, are available. In this course you will be using an open-source web application named Jupyter notebook maintained by Project Jupyter team. Jupyter's core supported programming languages are Julia, Python, and R. Jupyter ships with the IPython kernel in Jupyter allows you to write your programs in Python.

Visit this link to install Anancoda (then launch Jupyter notebook) on your personal computer. To start it on AUT machines follow below steps:

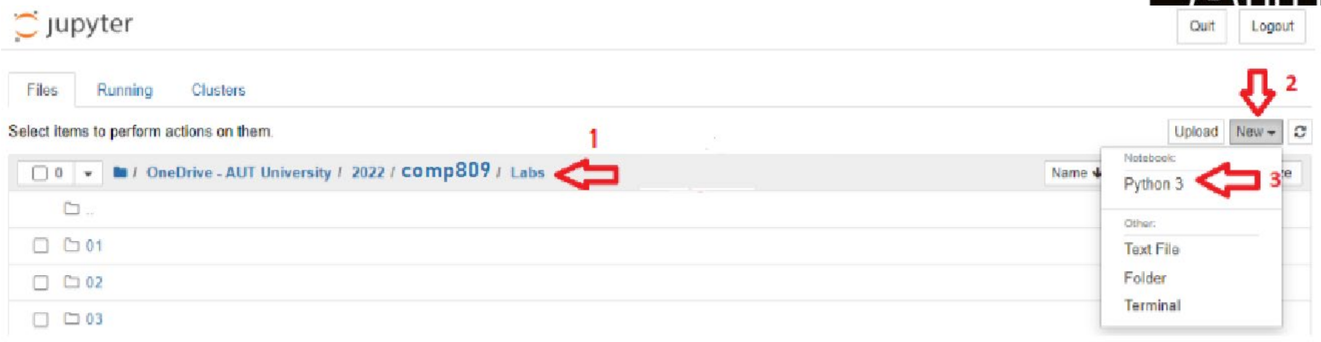
- Run Jupyter via the shortcut Anaconda adds to your start menu. A new tab will open in your default web browser, showing your Notebook Dashboard as below. Note that the default setting provides access only to the files and sub-folders contained within Jupyter's start-up directory. You can change the directory if required.



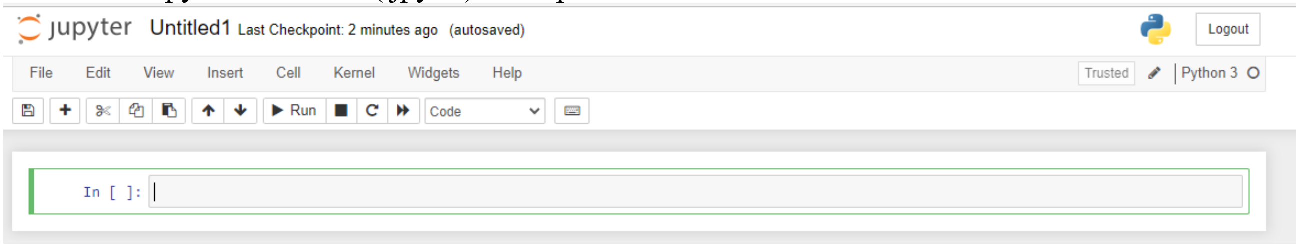
Select items to perform actions on them.

```
3D Objects              a year ago
Anaconda3               a year ago
Contacts                a year ago
Creative Cloud Files    a year ago
Desktop                 2 hours ago
```

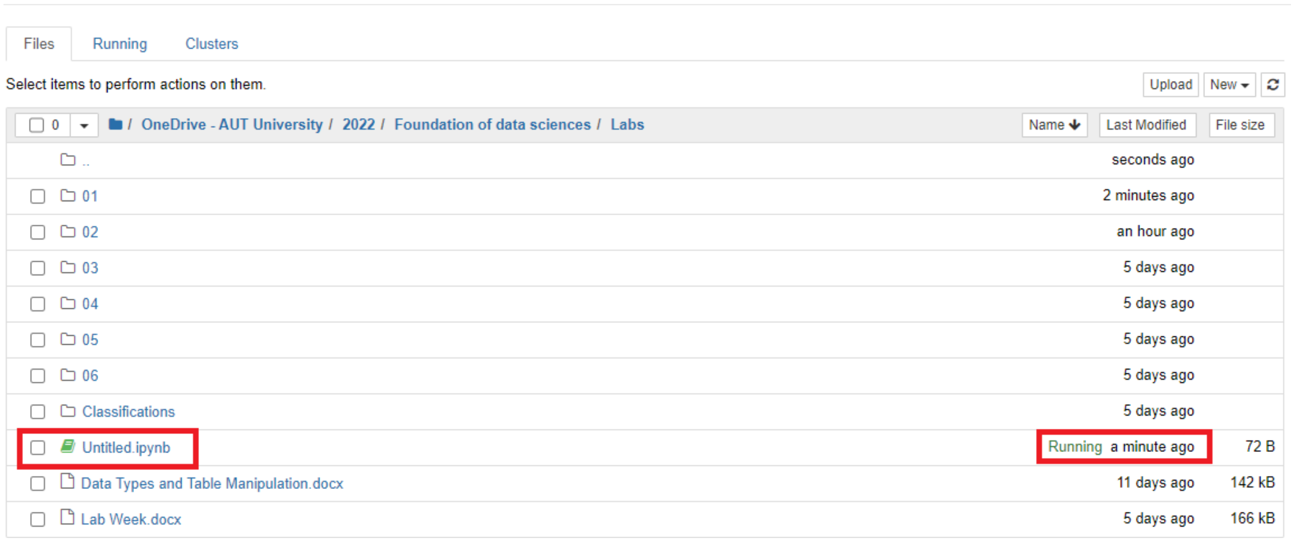
- Browse to the folder in which you would like to create your first notebook (e.g., I navigated to my Lab folder under my AUT's OneDrive), click the "New" drop-down button in the top-right and select "Python 3":



Your first Jupyter Notebook (.jpynb) will open in new tab:

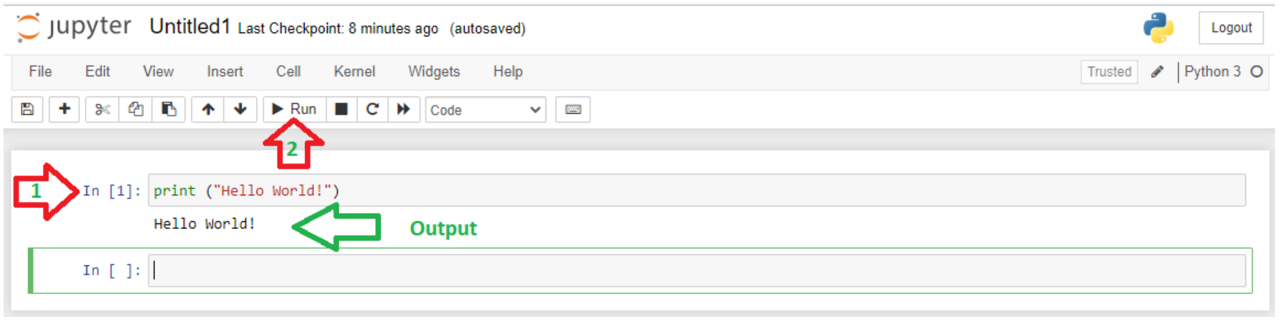


- Switch back to the dashboard (it is already opened in your browser). You will find your new Jupyter Notebook Untitled.ipynb created under your dashboard. Note the Running status showing the notebook is running:


Files
Running
Clusters
Select items to perform actions on them.
Upload
New
0
/ OneDrive - AUT University / 2022 / Foundation of data sciences / Labs
Name
Last Modified
File size

..	seconds ago	
01	2 minutes ago	
02	an hour ago	
03	5 days ago	
04	5 days ago	
05	5 days ago	
06	5 days ago	
Classifications	5 days ago	
Untitled.ipynb	Running a minute ago	72 B
Data Types and Table Manipulation.docx	11 days ago	142 kB
Lab Week.docx	5 days ago	166 kB

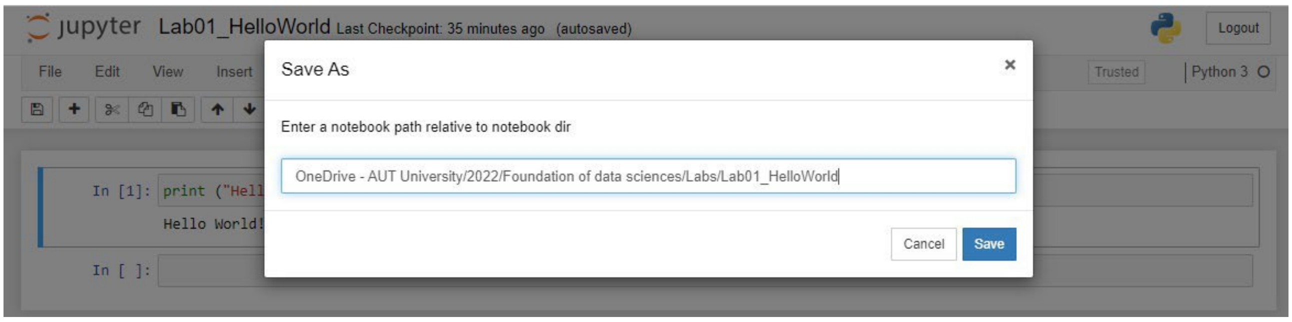
- Type print('Hello World!') in the code cell (the first cell in your notebook) and hit the Run button:



```
In [1]: print ("Hello World!")
        Hello World!

In [ ]:
```

- Click on the 'Save As' button under the 'File' and give a proper name to your notebook file and hit the “Save’ button:


jupyter Lab01_HelloWorld Last Checkpoint: 35 minutes ago (autosaved)
Logout
File Edit View Insert
Trusted
Python 3
Save As
Cancel
Save

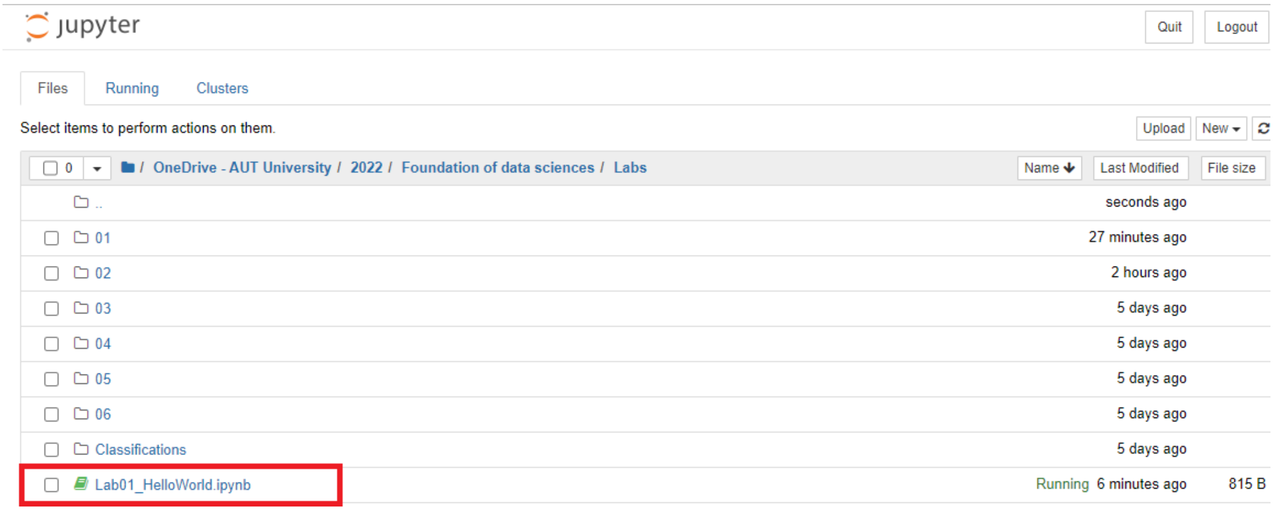
```
In [1]: print ("Hell
        Hello World!
In [ ]:
```

Enter a notebook path relative to notebook dir

OneDrive - AUT University/2022/Foundation of data sciences/Labs/Lab01_HelloWorld

Note: ‘%20’ indicates space in your folder’s name. To successfully save your file, replaced them with ‘ ‘ to match with the correct path.

- Switch back to the dashboard and note your notebook file with correct name:



Select items to perform actions on them.

/ OneDrive - AUT University / 2022 / Foundation of data sciences / Labs

	Name	Last Modified	File size
	..	seconds ago	
	01	27 minutes ago	
	02	2 hours ago	
	03	5 days ago	
	04	5 days ago	
	05	5 days ago	
	06	5 days ago	
	Classifications	5 days ago	
	Lab01_HelloWorld.ipynb	Running 6 minutes ago	815 B

3. Python Libraries and modules:

Python uses minimum libraries hence less memory in order to fast and efficientrunning of the program. We need to install and import libraries as we work. To do so use pip installer to install libraries:

```
pip install pandas # panads is one of the python’s open-source libraries
#for analysis and manipulation
```

Type the following code segments in your file and run your code to see the output.You are expected to get and understanding about syntax of python and basic concepts of python.

1. Basic Arithmetic Operations

```
print(2 + 2)
print(50 - 5*6)
print((50 - 5*6) / 4)
print(17%3) # the % operator returns the remainder of the division
print(2 ** 7) # 2 to the power of 7
```

2. Assignment of values to variables

```
width = 20
height = 5 * 9
print (width * height)
```

3. String Operations

```
print('spam eggs') # single quotes
print(r'C:\some\name') # note the r before the quote
print(3 * 'un' + 'ium')
```

Strings can be *indexed* (subscripted), with the first character having index 0. There is noseparate character type; a character is simply a string of size one:

```
word = 'Python'
print(word[0]) # character in position 0
print(word[5]) # character in position 5
```

Indices may also be negative numbers, to start counting from the right:

```
print(word[-1])# Last character
print(word[-2]) # Second-last character
print(word[-6])
print('J' + word[1:])
```

4. Lists

```
squares = [1, 4, 9, 16, 25]
print(squares)
```

Like strings (and all other built-in sequence types), lists can be indexed and sliced:

```
print(squares[0]) # indexing returns the item
print(squares[-1])
print(squares[-3:]) # slicing returns a new list
```

Lists also support operations like concatenation:

```
print(squares + [36, 49, 64, 81, 100])
```

Unlike strings, which are immutable, lists are a mutable type, i.e. it is possible to change theircontent:

```
cubes = [1, 8, 27, 65, 125]  # something's wrong here
4 ** 3                        # the cube of 4 is 64, not 65!
cubes[3] = 64                # replace the wrong value
print(cubes)
```

You can also add new items at the end of the list, by using the append() *method*

```
cubes.append(216) # add the cube of 6
cubes.append(7 ** 3) # and the cube of 7
print(cubes)
```

5. Loop in Python

Example: A Fibonacci sequence is the integer sequence of 0, 1, 1, 2, 3, 5… . The first two terms are 0 and 1. All other terms are obtained by adding the preceding two terms. This means to say the n^{th} term is the sum of $(n-1)^{th}$ and $(n-2)^{th}$ term. In other words, the sum of two elements defines the next.

```
 a, b = 0, 1
 while a < 10:
  print(a) # Note and respect the indent
  a, b = b, a+b # Note and respect the indent
```

Experiment with the following:

i) Instead of the single statement (a, b = b, a+b) use two different statements and examine the output.
ii) Explain why the result different from the original output.
iii) Modify ii) in order to obtain the original output.

6. If statements

```
x = int(input("Please enter an integer: "))
if x < 0:
 print('Negative value is changed to zero')
if x == 0:
 print("You chose '0'.")
elif x == 10:
 print("You chose '10'.")
elif x == 100:
 print("You chose '100'.")
else:
 print("Invalid choice.")
```

7. For statement

```
# Measure strings:
```

```
words = ['cat', 'window', 'defenestrate']
for x in words:
 print(x, len(x))
```

If you do need to iterate over a sequence of numbers, the built-in function range() comes inhandy. It generates arithmetic progressions:

```
for i in range(5):
 print(i)
```

8. Defining Functions:

8.1. **Example1:** Create a function that uses while loop to print the integer numbers up to certain limit:

```
def Mynumbers(limit):
    i = 0
    numbers = []

    while i < limit:
        numbers.append(i)
        i = i + 1
    print (numbers)
```

To execute the function, you can either

○ call the function and hard code the 'limit' value:

```
Mynumbers(25)
```
# Prints numbers up to 25

- Or pass the value from keyboard:

```
limit = int(input("Please enter an integer: "))
Mynumbers(limit)
```

```
Please enter an integer: 25
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24]
```

8.2. **Example2:** Work with Fibonacci Sequence

Below function prints the nth Fibonacci number:

```
def Myfibo(n):
   if n <= 1:
       return n
   else:
       return(Myfibo(n-1) + Myfibo(n-2))

fiboNum = int(input("Please enter an integer: "))

# check if the number of terms is valid
if fiboNum <= 0:
   print("Number must be a positive integer!")
else:
   print("Fibonacci sequence:")
   for i in range(fiboNum):
       print(Myfibo(i))
```

9. To Do:

9.1. Write a function (Fib()) to get a value (*n*) and print a series (following below example's logic) **UP TO** value *n*.

For example:

```
Fib(2000)
```

```
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597
```

9.2. Write a function that returns either a string 'Found' or 'NotFound' depending on whether or not the input number is located in a list of numbers of a given size (say 10).