

## **Introduction to Jupyter & Python:**

We will need Skimage, NumPy, Jupiter and other libraries, most of them are included in Anaconda.

Anaconda includes a package manager Conda and hundreds of scientific packages.

### **Installation:**

On Windows: Just install Anaconda.

<https://conda.io/docs/user-guide/install/windows.html>

On Linux:

<https://conda.io/docs/user-guide/install/linux.html>

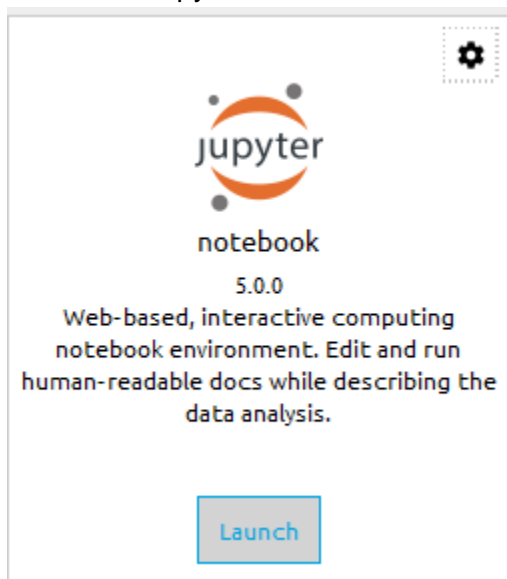
<https://conda.io/docs/user-guide/install/test-installation.html>

Attention: Anaconda needs around 3 GB for installation and takes a considerable time to install!

Working with Notebooks:

1- Open Anaconda Navigator

2- Launch Jupyter Notebook



A windows similar to the following will appear:

Files

Running

Clusters

Select items to perform actions on them.

Anaconda2

Contacts

Desktop

Documents

Downloads

Favorites

Links

Music

Pictures

Saved Games

Searches

Videos

3- Browse to the directory that contains the notebook (click on the name of the directory to open it).

/ Desktop / IPLabIntroToPythonJupyter / std version

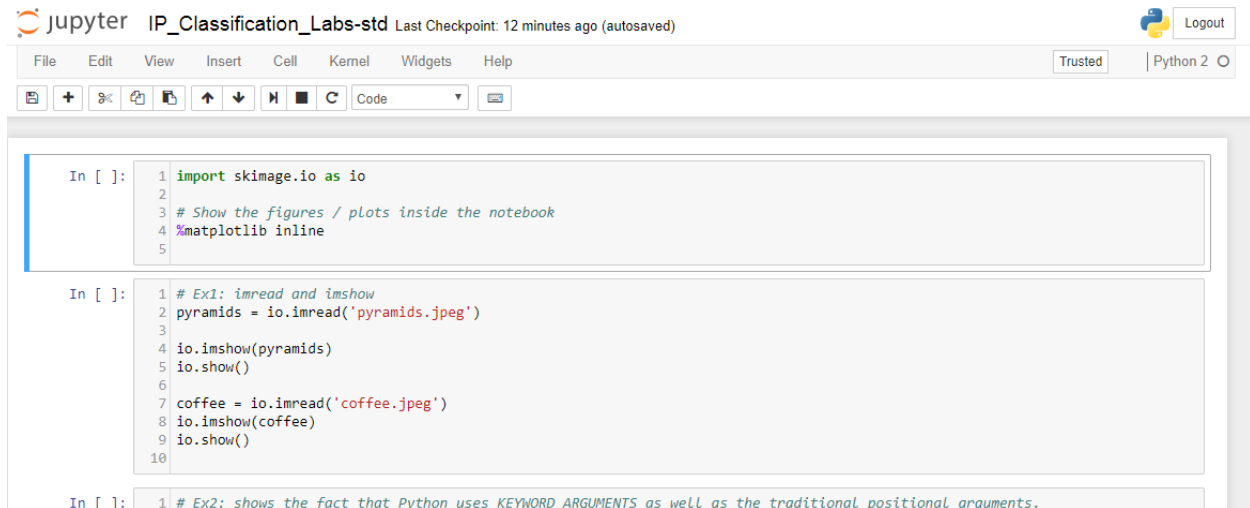
..

IP\_Classification\_Labs-std.ipynb

coffee.jpeg

pyramids.jpeg


4- Press on the required notebook, something similar to the following will appear:



```
In [ ]: 1 import skimage.io as io
2
3 # Show the figures / plots inside the notebook
4 %matplotlib inline
5

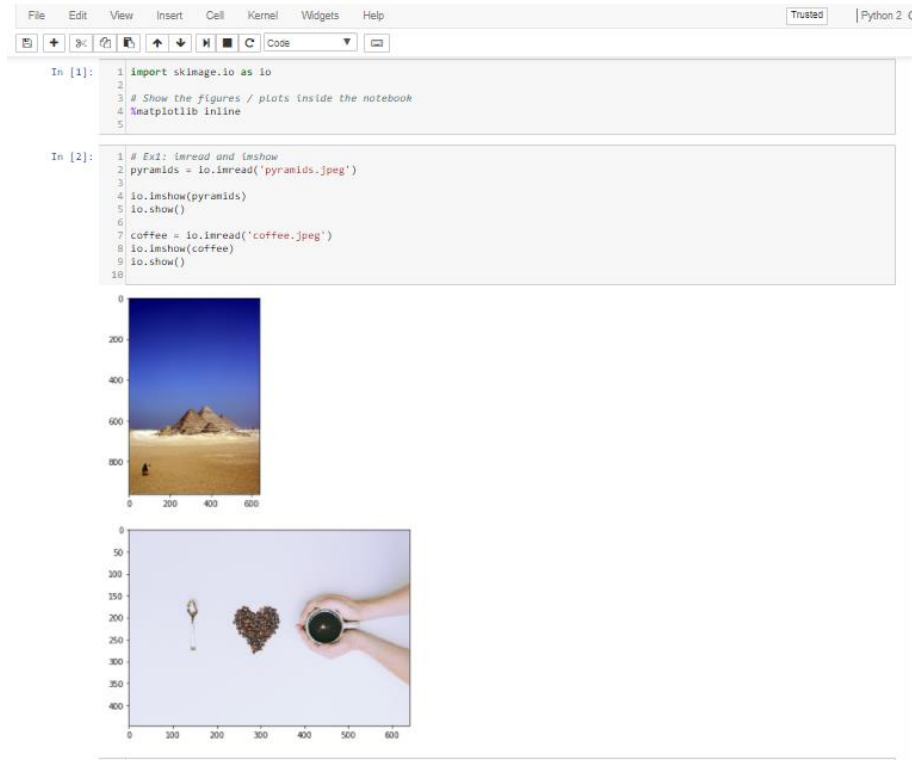
In [ ]: 1 # Ex1: imread and imshow
2 pyramids = io.imread('pyramids.jpeg')
3
4 io.imshow(pyramids)
5 io.show()
6
7 coffee = io.imread('coffee.jpeg')
8 io.imshow(coffee)
9 io.show()
10

In [ ]: 1 # Ex2: shows the fact that Python uses KEYWORD ARGUMENTS as well as the traditional positional arguments.
```

To execute any cell click inside the cell and then press the button (  ). Each part grey level part is called cell and can be executed separately.

**Testing on our notebook -required before lab time:**

Execute the first 3 cells. The output should be as follows:



# Intro To Python

To open jupyter notebook use Anaconda Navigator (installed with anaconda) and Press Launch on Jupyter.

## Functions in python

One of the great features of Python is the ability to use KEYWORD ARGUMENTS as well as the traditional positional arguments.

For example:

The function:

---

```
def fun(x,y,z):  
    print x,y,z
```

---

Can be called as:

---

```
fun(1,2,3); #Or  
fun(z=3,y=2,x=1)
```

---

In both cases the output will be: 1 2 3

See Example 2 in the notebook

## Reading and showing images

---

```
import skimage.io as io  
# to read image;  
Img = io.imread('image.png')  
# To show image  
io.imshow(img)  
io.show()
```

---

## Dealing with arrays

Numpy is a great mathematical library that deals with array.

To construct array of zeros [5,5,5] all of unit8 (range from 0 to 255)

---

```
Import numpy as np  
Arr = np.zeros(shape=(5,5,5),dtype=np.uint8)
```

---

## **Array Indexing**

Arrays can be indexed in a way similar to matlab, with two differences:

It uses square brackets and index starts with 0.

`Arr[0:50,0:50,1] = 20`

Will change the set of intersection of the first 50 rows and first 50 columns in column 1 to 20.

### **`np.copy`**

Is used to construct a copy of the sent array (matrix) and returns it.

`copiedArr = np.copy(Arr)`

## **Requirements**

Open attached notebooks to find requirements' details and hints

General Hints:

Filters References:

<http://scikit-image.org/docs/0.12.x/api/skimimage.filters.html>

ImNoise:

<http://scikit-image.org/docs/dev/api/skimimage.util.html>

Features (including Canny):

<http://scikit-image.org/docs/dev/api/skimimage.feature.html>

Equalize\_hist:

[http://scikit-image.org/docs/dev/api/skimimage.exposure.html#skimimage.exposure.equalize\\_hist](http://scikit-image.org/docs/dev/api/skimimage.exposure.html#skimimage.exposure.equalize_hist)

