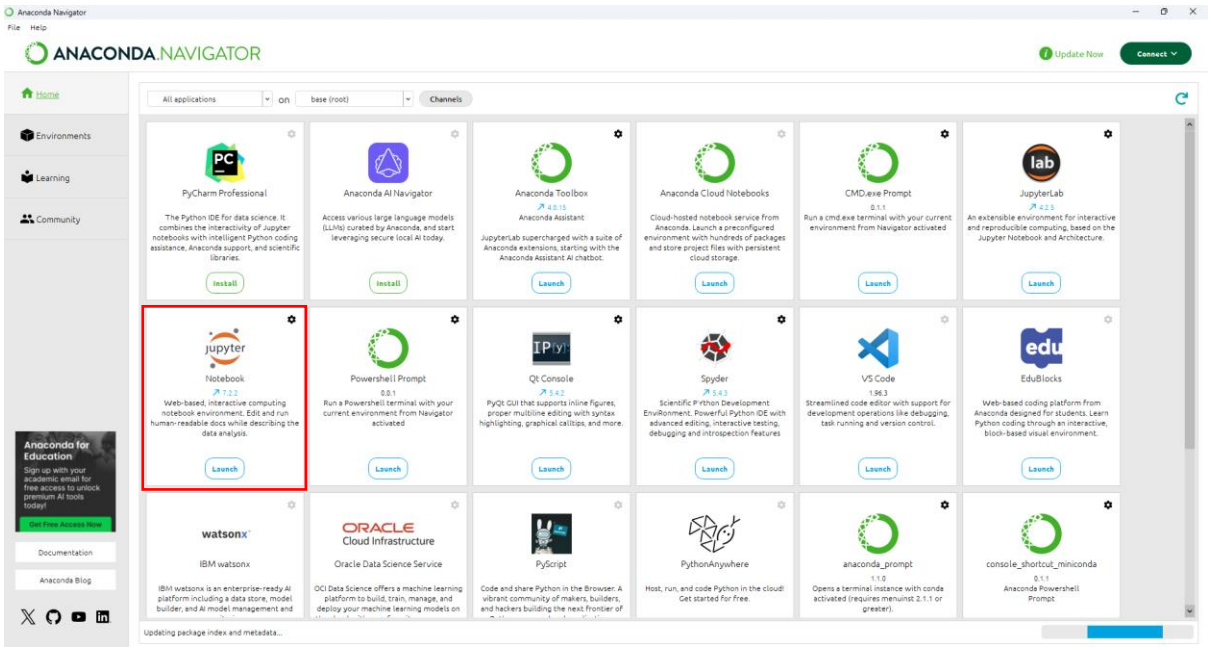
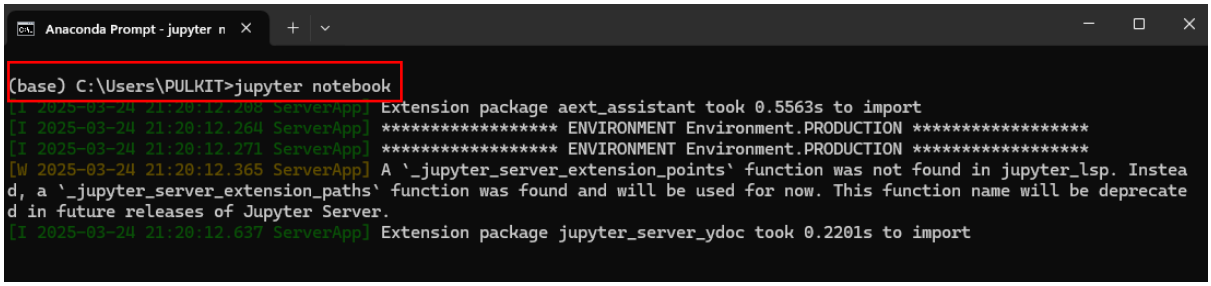
😊 Python Numbers and Variables

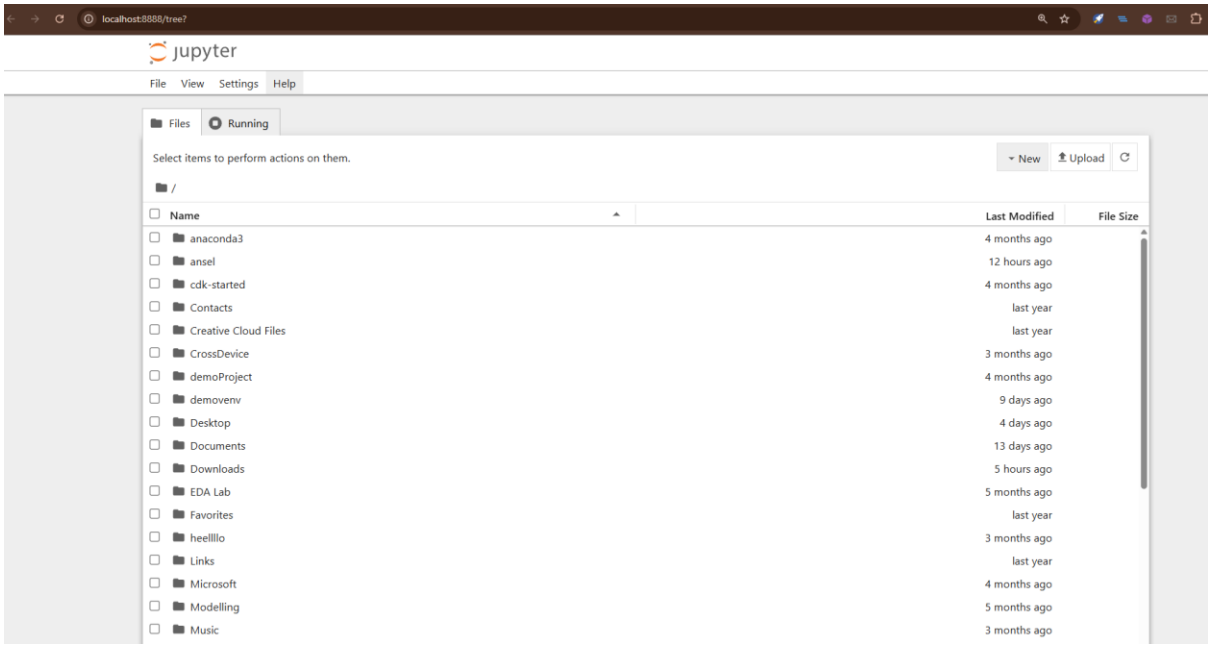
1. In this lab, we will learn about Python numbers and variables. All of the content related to Python will be available on this GitHub repository and you can clone this repository to get all the Python notebooks on your system.
2. Now let's open our first notebook, but before that, you should have Python on your system. You can download Python directly from the https://www.python.org/
3. Then you need to download Anaconda on your system using this link https://www.anaconda.com/
4. After installing Anaconda, you can either open the Navigator or the Prompt. In the navigator, you can look for Jupyter Notebook.



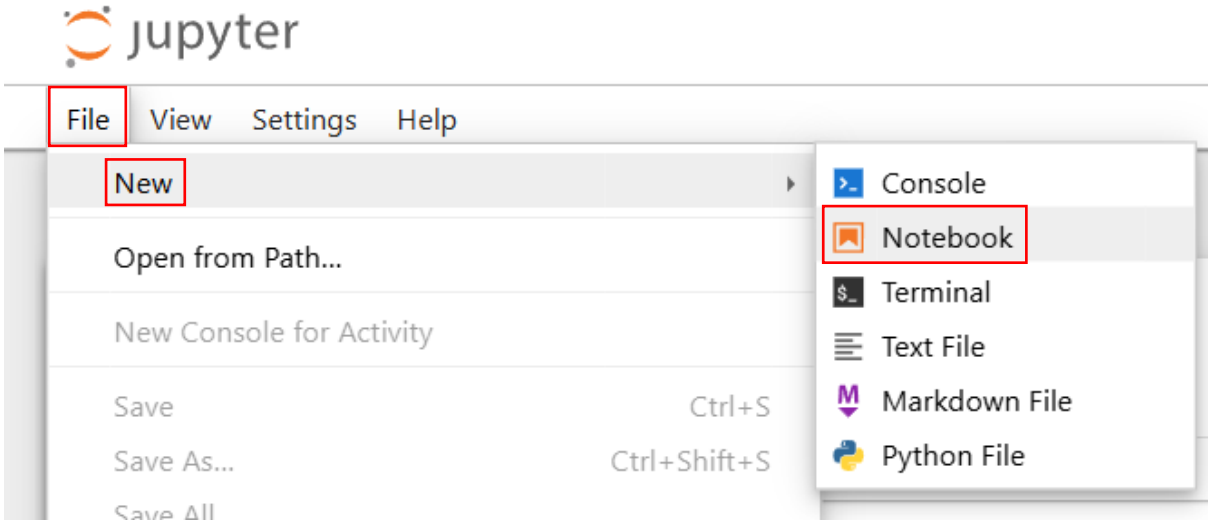
5. And in the anaconda prompt you can directly write **Jupyter notebook** and it will open the Notebook for you.



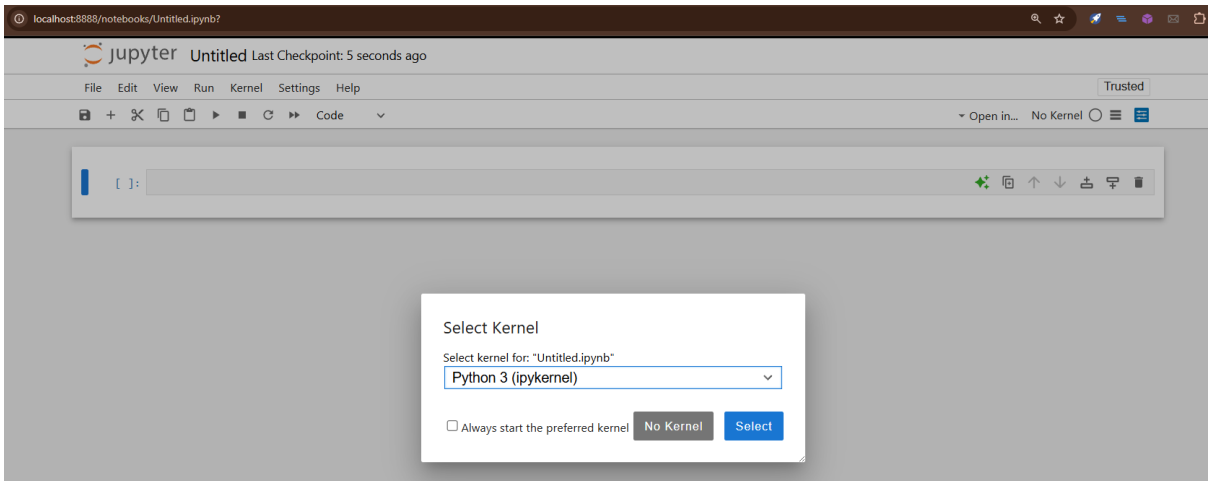
6. Once you are inside the notebook you can see some folders. These are the system folders of your local machine.



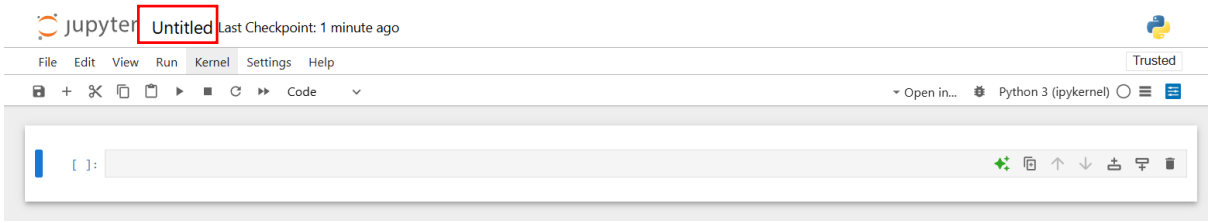
7. Now we will create our first notebook, click on **File >> New** then choose **Notebook**.

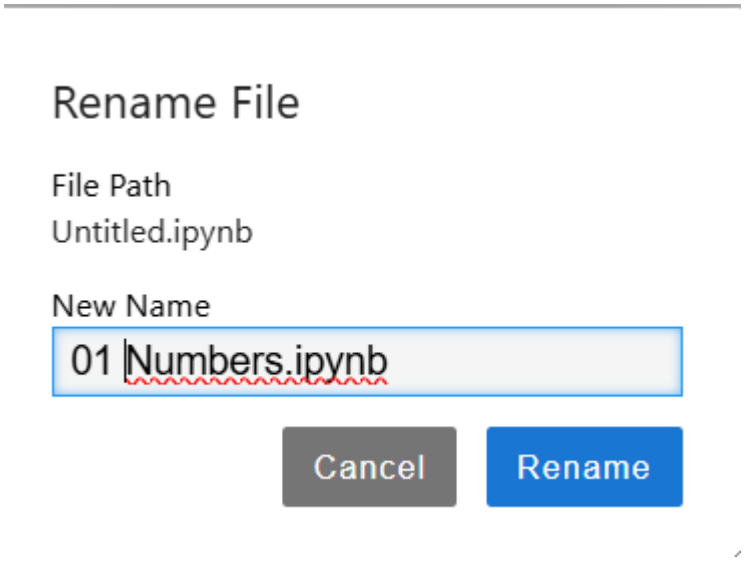


8. You will be directed to a new page, choose Python 3 as your kernel. Click on Select.

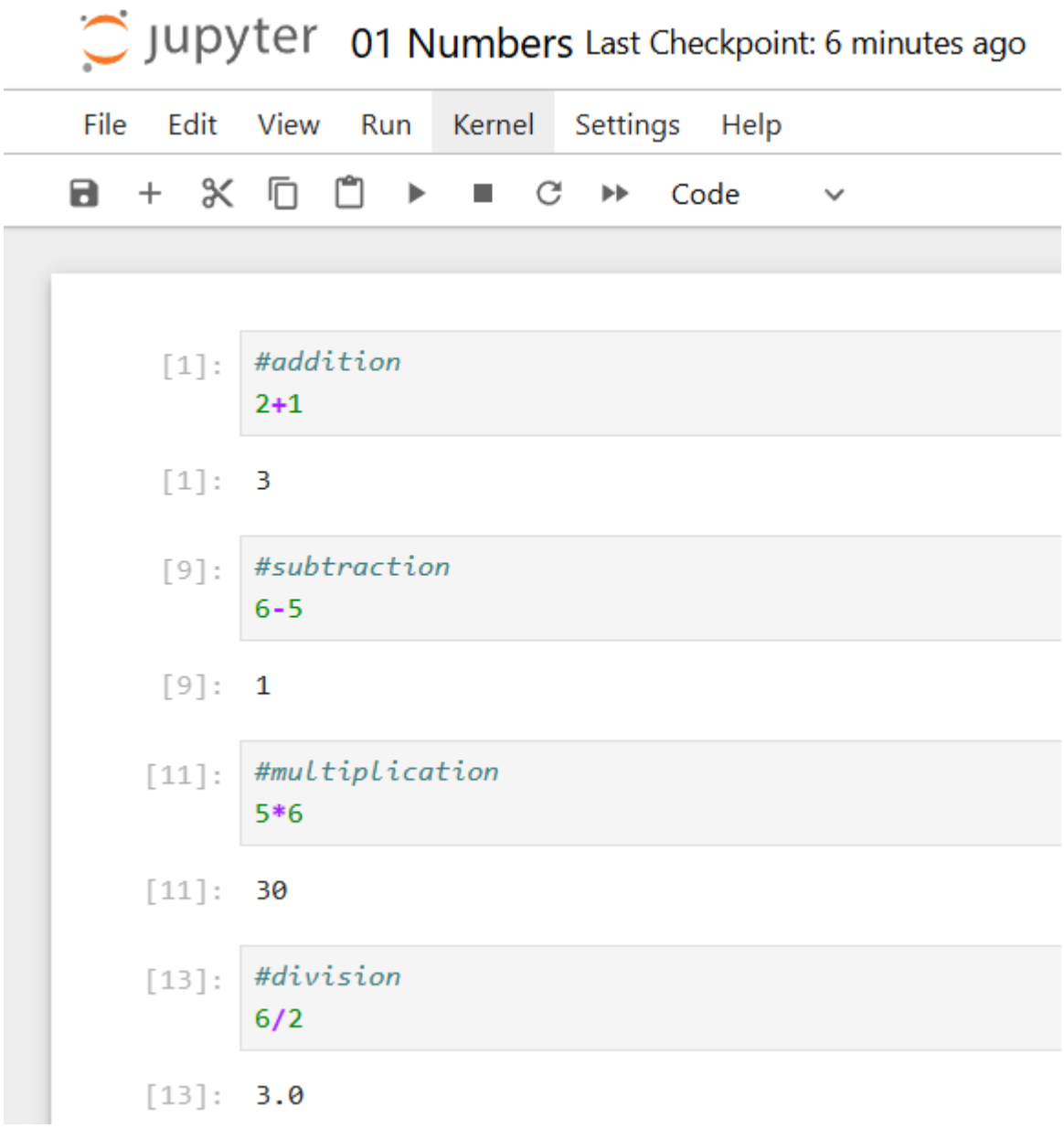


9. First, we will change the name of our notebook. Click on Untitled to change the title.





10. In the Python notebook, we will be doing some basic math operations like addition, subtraction, multiplication, and division.



11. Below you can see that when we divided 7 by 4, we get 1 this is called floor division. The // operator (two forward slashes) truncates the decimal without rounding and returns an integer result.

```
[17]: #floor division
      7//4
[17]: 1
```

12. Below what we did is called **modulo** means 4 goes into 7 once, with a remainder of 3. The % operator returns the remainder after division.

```
[19]: #Modulo
      7%4

[19]: 3
```

13. These are some more basic arithmetic operations that you see below in Python.

```
[21]: # Powers
      2**3

[21]: 8

[23]: # Can also do roots this way
      4**0.5

[23]: 2.0

[25]: # Order of Operations followed in Python
      2 + 10 * 10 + 3

[25]: 105

[27]: # Can use parentheses to specify orders
      (2+10) * (10+3)

[27]: 156
```

14. Now we are going to see some variable assignments, as you can see below, we said to Python notebook that the value of **a** is 5 then we added **a** with **a** and we got 10.

Variable Assignments

Now that we've seen how to use numbers in Python as a calculator let's see how we can assign names and create variables.

We use a single equals sign to assign labels to variables. Let's see a few examples of how we can do this.

```
[31]: # Let's create an object called "a" and assign it the number 5
      a = 5
```

Now if I call *a* in my Python script, Python will treat it as the number 5.

```
[34]: # Adding the objects
      a+a

[34]: 10
```

15. Let's see if we reassign the same number to the same variable as you can see yes we can reassign the same things.

What happens on reassignment? Will Python let us write it over?

[37]:
```
# Reassignment
a = 10
```

[39]:
```
# Check
a
```

[39]:
```
10
```

Yes! Python allows you to write over assigned variable names. We can also use the variables themselves when doing the reassignment. Here is an example of what I mean:

[42]:
```
# Check
a
```

[42]:
```
10
```

[44]:
```
# Use A to redefine A
a = a + a
```

[46]:
```
# Check
a
```

[46]:
```
20
```

16. We can use variable names as you can see below. Here we are using object names to keep track of things in a meaningful way.

Using variable names can be a very useful way to keep track of different variables in Python. For example:

[49]:
```
# Use object names to keep better track of what's going on in your code!
my_income = 100

tax_rate = 0.1

my_taxes = my_income*tax_rate
```

[51]:
```
# Show my taxes!
my_taxes
```

[51]:
```
10.0
```