



**Chan  
Zuckerberg  
Initiative** 



**Marcelo Leomil Zoccoler**

With materials from Robert Haase

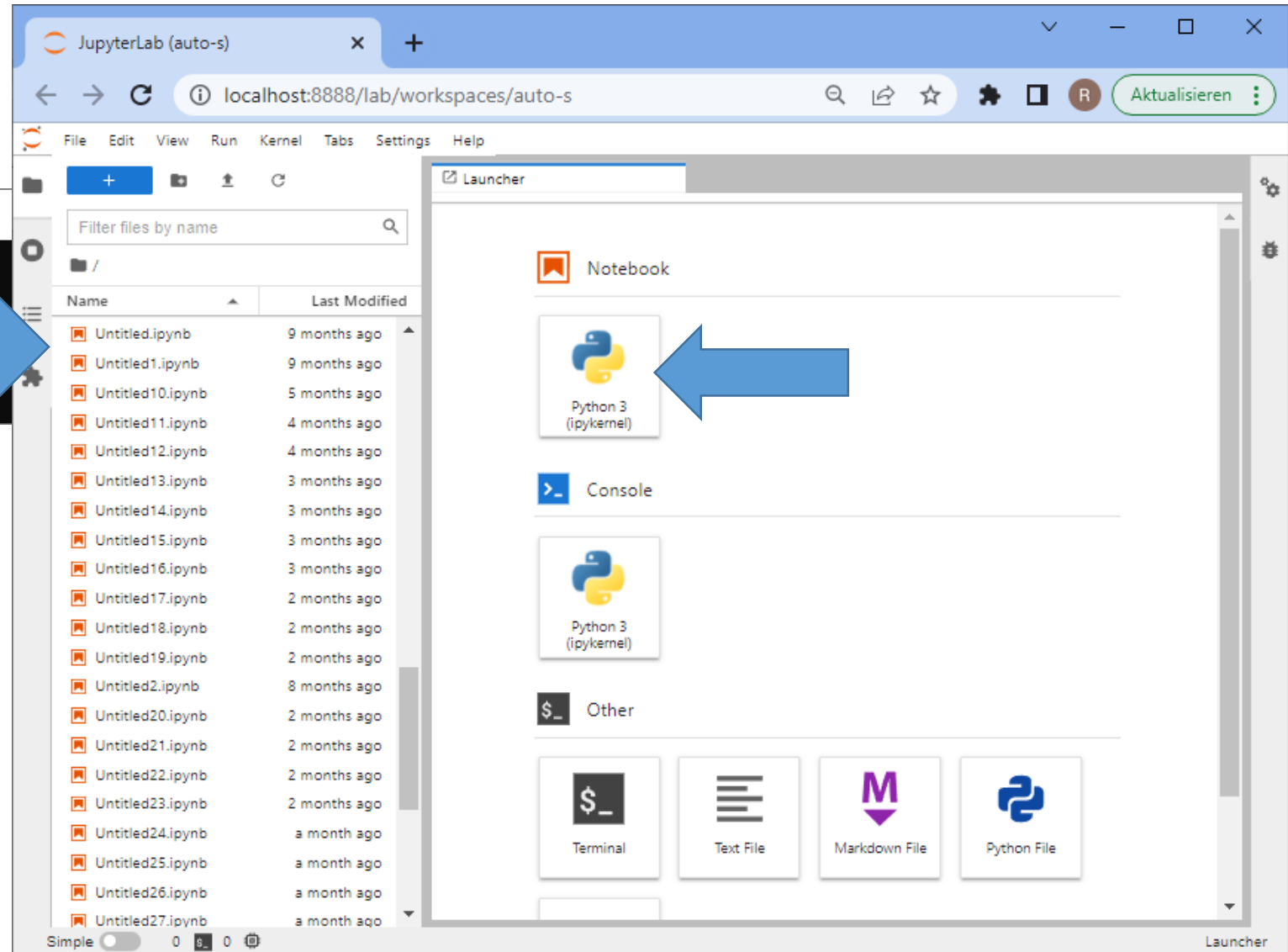
August 2023

- Our programming environment for this course

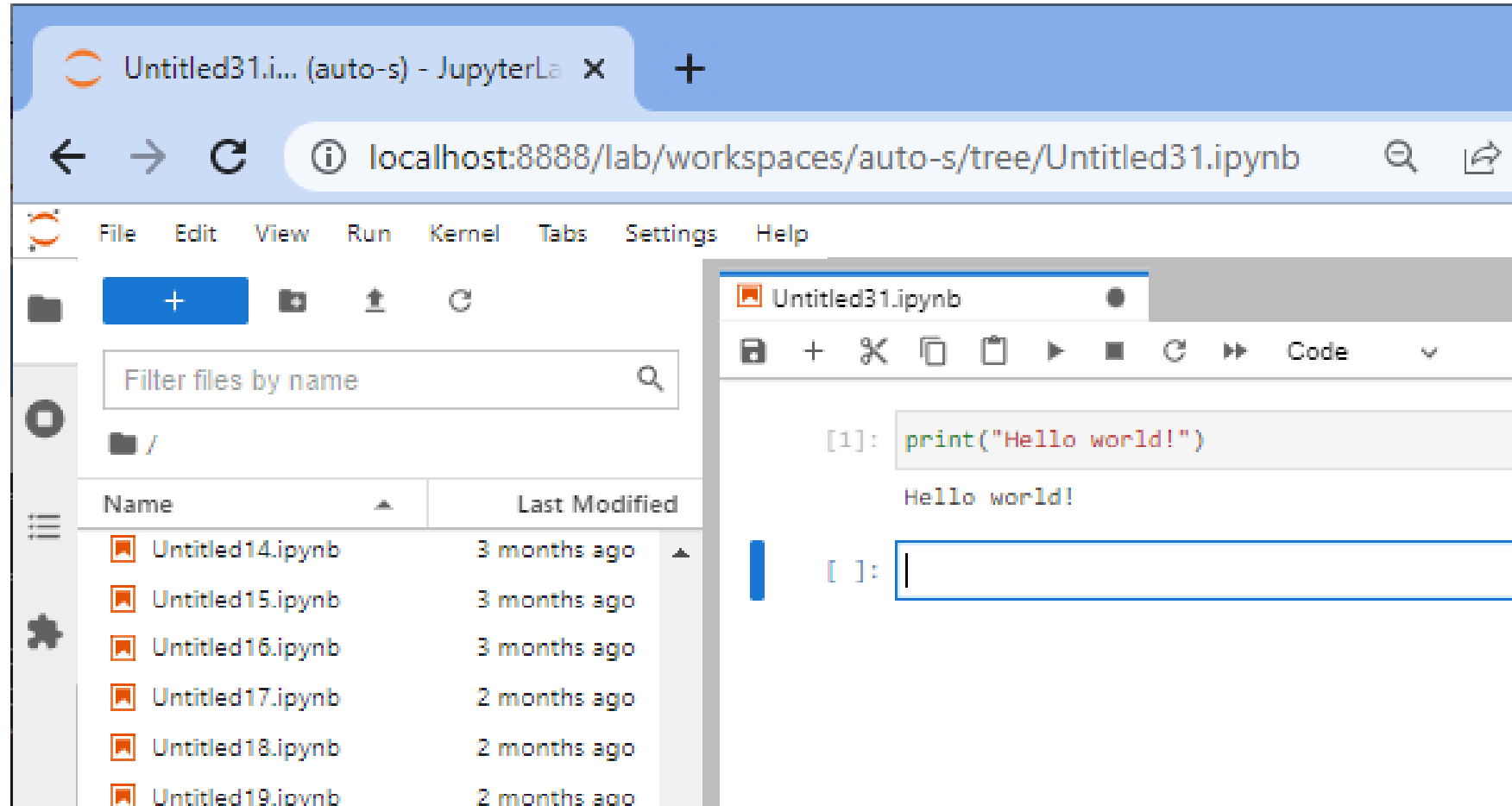
```
Miniforge Prompt - mamba deactivate  
  
(base) C:\Users\mazo260d>mamba activate napari-latam  
(napari-latam) C:\Users\mazo260d>jupyter lab
```

Current environment

Current path



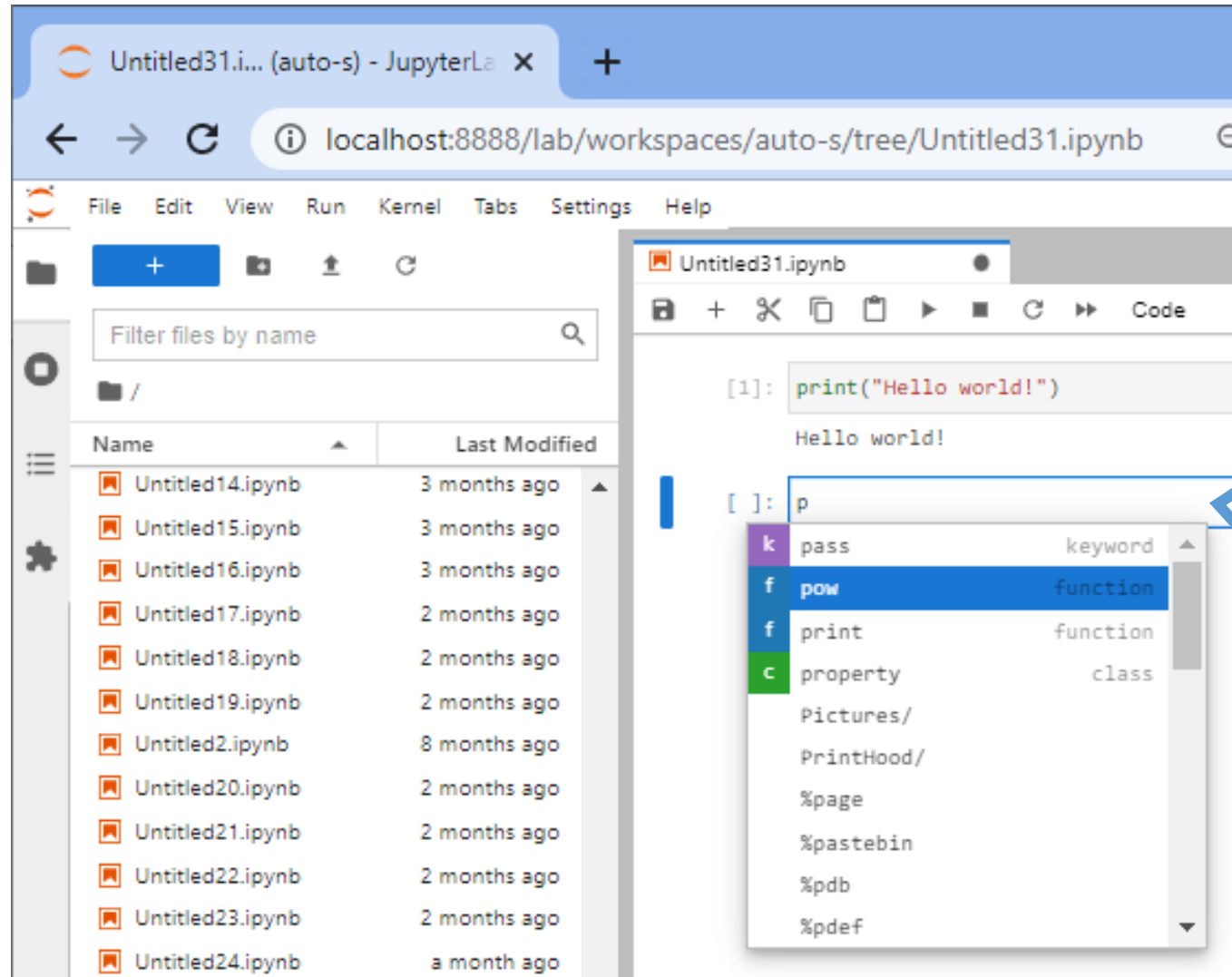
- Execute code cell-by-cell and see results instantaneously



The screenshot shows the JupyterLab web interface. The top bar displays the browser tab 'Untitled31.i... (auto-s) - JupyterLa' and the URL 'localhost:8888/lab/workspaces/auto-s/tree/Untitled31.ipynb'. The left sidebar contains a file browser with a search bar 'Filter files by name' and a list of files: 'Untitled14.ipynb', 'Untitled15.ipynb', 'Untitled16.ipynb', 'Untitled17.ipynb', 'Untitled18.ipynb', and 'Untitled19.ipynb'. The main area shows the 'Untitled31.ipynb' file open in a code editor. The code cell contains the text `[1]: print("Hello world!")` and its output 'Hello world!'. A blue box highlights the code cell, and a callout bubble points to it with the text 'SHIFT + ENTER to execute a code cell'.

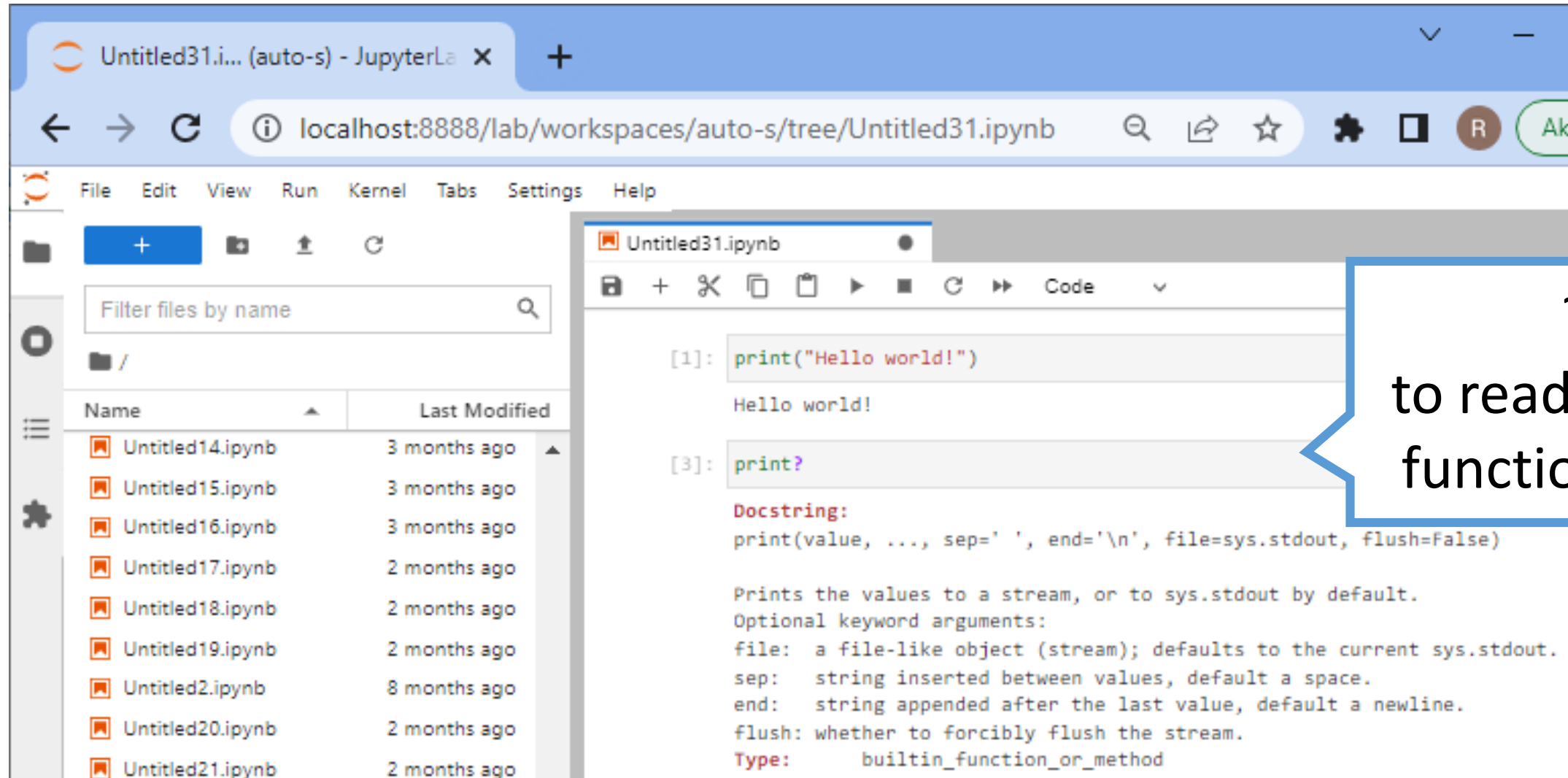
SHIFT + ENTER  
to execute a  
code cell

- Context-specific help, auto-completion



TAB  
to open auto-  
completion

- Help / “docstrings”



The screenshot shows the JupyterLab interface. On the left is a file browser with a search bar and a list of files. The main area on the right is the code editor, which shows the following code:

```
[1]: print("Hello world!")  
Hello world!  
  
[3]: print?
```

Below the code, the docstring for the `print` function is displayed:

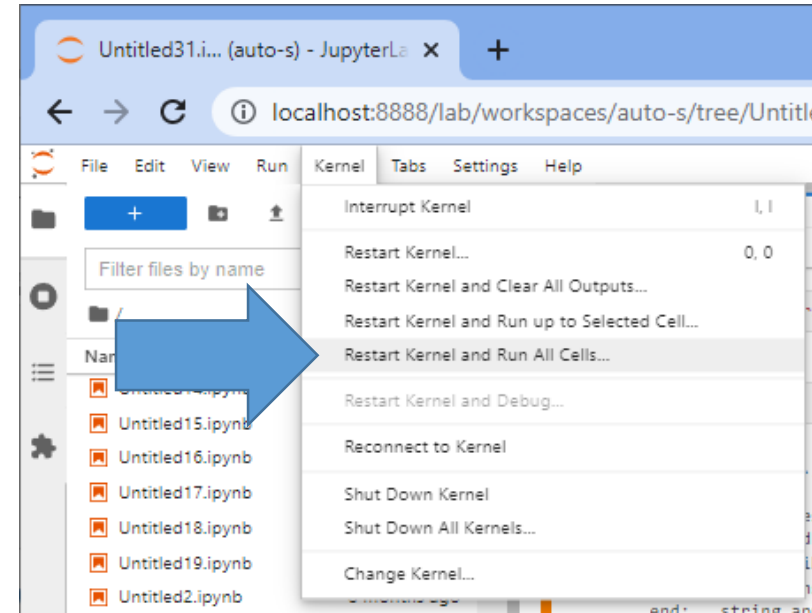
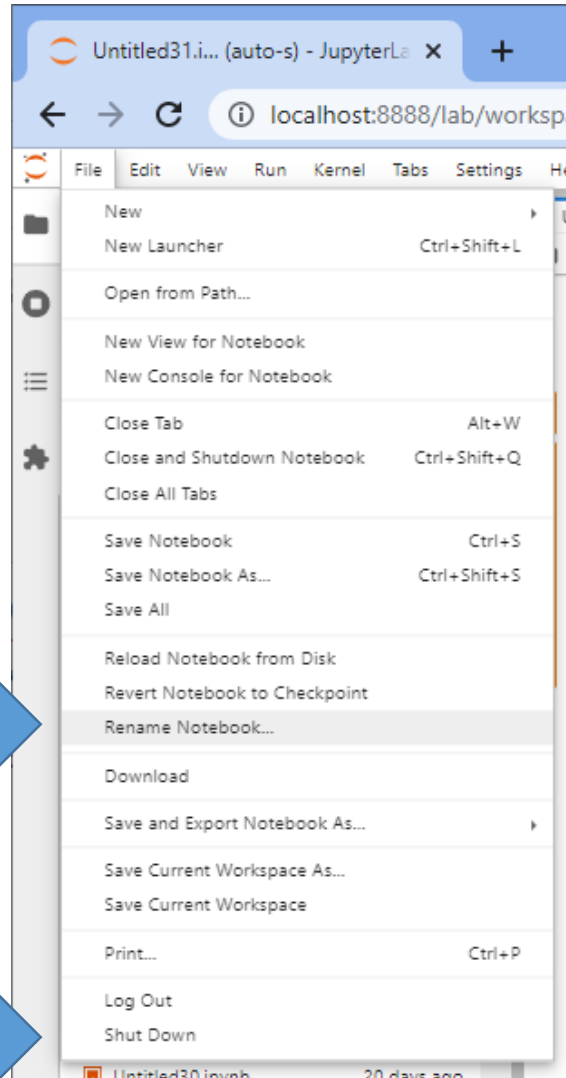
**Docstring:**  
`print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)`

Prints the values to a stream, or to `sys.stdout` by default.  
Optional keyword arguments:  
`file`: a file-like object (stream); defaults to the current `sys.stdout`.  
`sep`: string inserted between values, default a space.  
`end`: string appended after the last value, default a newline.  
`flush`: whether to forcibly flush the stream.

**Type:** builtin\_function\_or\_method

A callout box with a blue border and a question mark icon contains the text: "? to read what a function does".

- Saving / renaming / closing



Enforcing a “clean” execution state is important for ensuring reproducibility and repeatability



**Chan  
Zuckerberg  
Initiative** 

# Python programming basics

Marcelo Leomil Zoccoler

With materials from Robert Haase

August 2023

- Variables can hold numeric values and you can do math with them

```
▶ # initialize program  
a = 5  
b = 3  
  
# run algorithm on given parameters  
sum = a + b  
  
# print out result  
print (sum)
```

8



- Math commands supplement operators to be able to implement any form of calculations

- Power

```
▶ pow(3, 2)
```

```
] : 9
```

- Absolute

```
▶ abs(-8)
```

```
] : 8
```

- Rounding

```
▶ round(4.6)
```

```
] : 5
```

Be careful with  
some of them!

```
▶ round(4.5)
```

```
] : 4
```

[https://en.wikipedia.org/wiki/Rounding#Round\\_half\\_to\\_even](https://en.wikipedia.org/wiki/Rounding#Round_half_to_even)

Comments should contain additional information such as

- User documentation
  - What does the program do?
  - How can this program be used?
- Your name / institute in case a reader has a question
- Comment why things are done.
- Do not comment what is written in the code already!

```
#  
# This program sums up two numbers.  
#  
# Usage:  
# * Run it in Python 3.8  
#  
# Author: Robert Haase, PoL TUD  
#         Robert.haase@tu-dresden.de  
# April 2021  
  
# initialise program  
a = 1  
b = 2.5  
  
# run complicated algorithm  
final_result = a + b  
  
# print the final result  
print( final_result )
```

- Also strings as values for variables are supported

Single and double quotes  
allowed

```
▶ firstname = "Robert"  
  lastname = 'Haase'  
  
print("Hello " + firstname + " " + lastname)
```

Hello Robert Haase

- String **f**ormatting is made easy using f-strings.

```
f"This is an f-string. a's value is {a}. Doubling the value of a gives {2*a}."
```

```
"This is an f-string. a's value is 5. Doubling the value of a gives 10."
```

- Using f-strings, you can also call code from within a string. Take care of code readability!

```
f"The first_name variable contains {first_name.lower().count('r')} r letters."
```

```
'The first_name variable contains 2 r letters.'
```

- Also strings as values for variables are supported
- When combining strings and numbers, you need to explicitly define what you want to do.

```
❏ # mixing types

a = 5
b = "2"

print (a + b)
```

---

```
TypeError                                 Traceback (most recent call last)
<ipython-input-4-51629e6a285f> in <module>
      4 b = "2"
      5
----> 6 print (a + b)

TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

```
❏ # mixing types to make numbers

a = 5
b = "2"

print (a + int(b))
```

7

```
❏ # mixing types

a = "5"
b = 2

print (a + b)
```

---

```
TypeError                                 Traceback (most recent call last)
<ipython-input-5-85ae49867097> in <module>
      4 b = 2
      5
----> 6 print (a + b)

TypeError: can only concatenate str (not "int") to str
```

```
❏ # mixing types to make strings

a = "5"
b = 2

print (a + str(b))
```

52

- Conversion to a floating point number: float()