

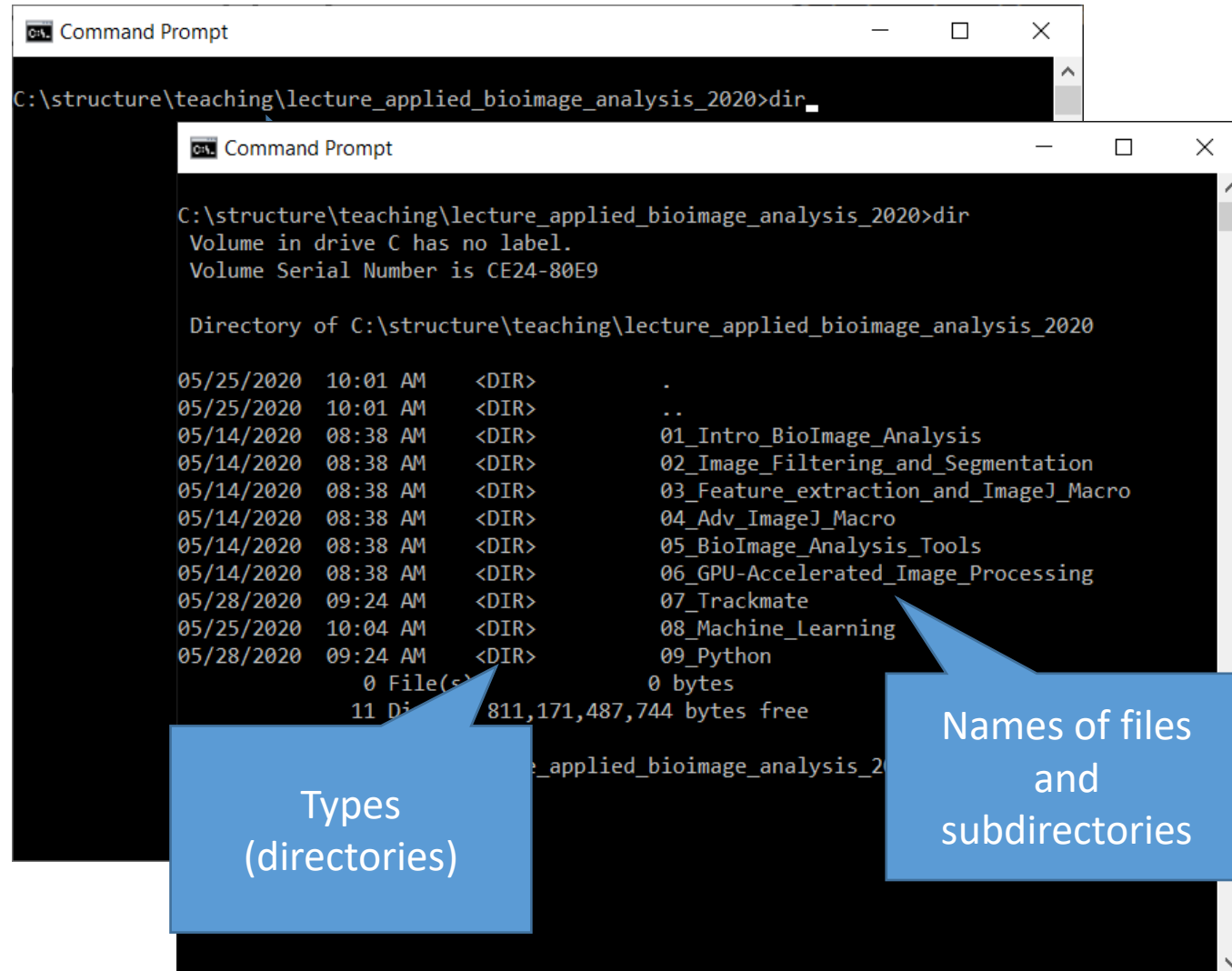
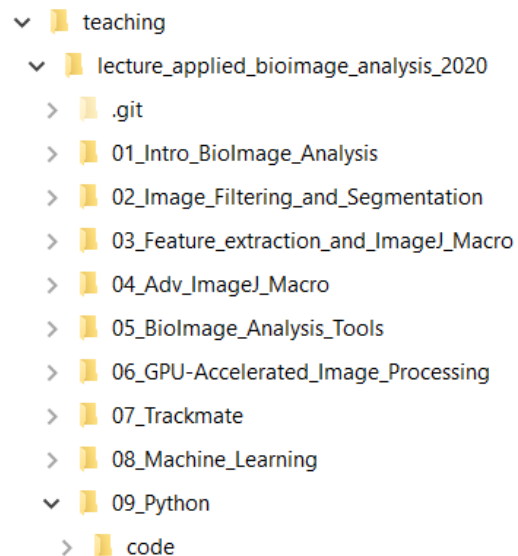
# The command line

Robert Haase

October 2022

# The command line

- A.k.a. the Terminal or Eingabeaufforderung: Welcome to the 20<sup>th</sup> century!
- The `dir` command tells you what's in the current directory
- On Mac and Linux the command is called `ls -l`



```
C:\structure\teaching\lecture_applied_bioimage_analysis_2020>dir

C:\structure\teaching\lecture_applied_bioimage_analysis_2020>dir
Volume in drive C has no label.
Volume Serial Number is CE24-80E9

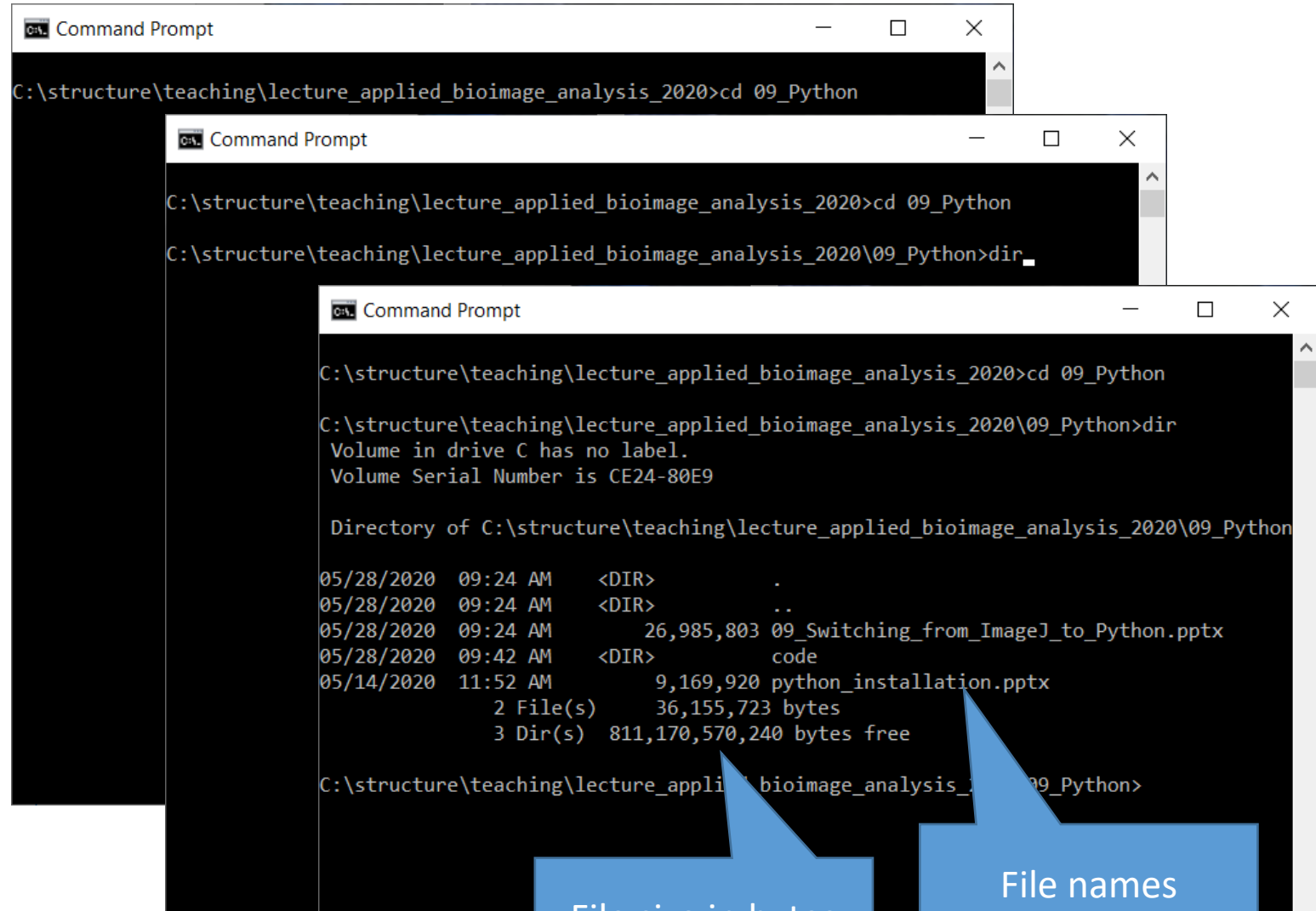
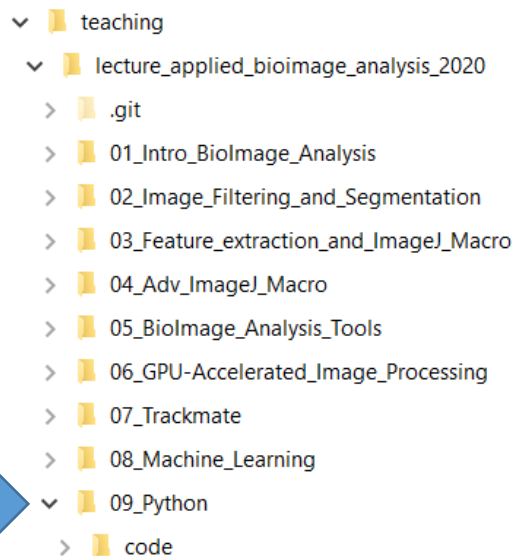
Directory of C:\structure\teaching\lecture_applied_bioimage_analysis_2020

05/25/2020  10:01 AM    <DIR>          .
05/25/2020  10:01 AM    <DIR>          ..
05/14/2020  08:38 AM    <DIR>          01_Intro_BioImage_Analysis
05/14/2020  08:38 AM    <DIR>          02_Image_Filtering_and_Segmentation
05/14/2020  08:38 AM    <DIR>          03_Feature_extraction_and_ImageJ_Macro
05/14/2020  08:38 AM    <DIR>          04_Adv_ImageJ_Macro
05/14/2020  08:38 AM    <DIR>          05_BioImage_Analysis_Tools
05/14/2020  08:38 AM    <DIR>          06_GPU-Accelerated_Image_Processing
05/28/2020  09:24 AM    <DIR>          07_Trackmate
05/25/2020  10:04 AM    <DIR>          08_Machine_Learning
05/28/2020  09:24 AM    <DIR>          09_Python
               0 File(s)              0 bytes
               11 Dir(s)          811,171,487,744 bytes free
```

Types (directories)

Names of files and subdirectories

- A.k.a. the Terminal or Eingabeaufforderung: Welcome to the 20<sup>th</sup> century!
- The `cd` command let's you move between different directories.
- With `cd <pathname>` you go into a sub-directory



```
Command Prompt
C:\structure\teaching\lecture_applied_bioimage_analysis_2020>cd 09_Python

Command Prompt
C:\structure\teaching\lecture_applied_bioimage_analysis_2020>cd 09_Python
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python>dir

Command Prompt
C:\structure\teaching\lecture_applied_bioimage_analysis_2020>cd 09_Python
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python>dir
Volume in drive C has no label.
Volume Serial Number is CE24-80E9

Directory of C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python

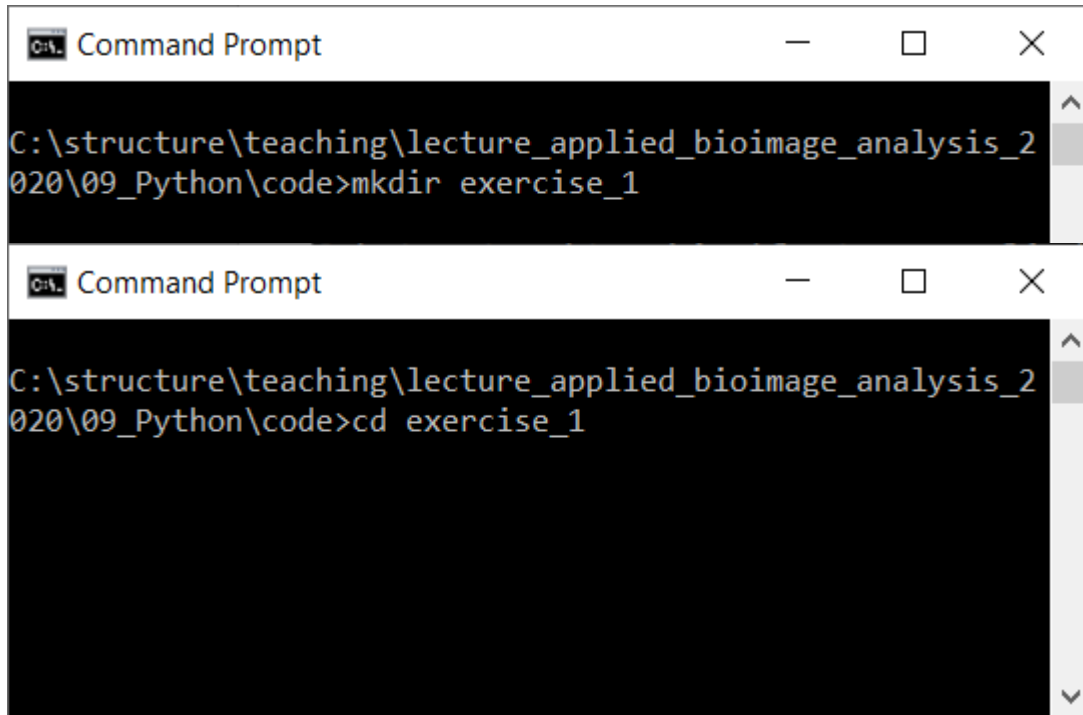
05/28/2020  09:24 AM    <DIR>          .
05/28/2020  09:24 AM    <DIR>          ..
05/28/2020  09:24 AM             26,985,803  09_Switching_from_ImageJ_to_Python.pptx
05/28/2020  09:42 AM    <DIR>          code
05/14/2020  11:52 AM             9,169,920 python_installation.pptx
               2 File(s)          36,155,723 bytes
               3 Dir(s)      811,170,570,240 bytes free

C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python>
```

File size in bytes

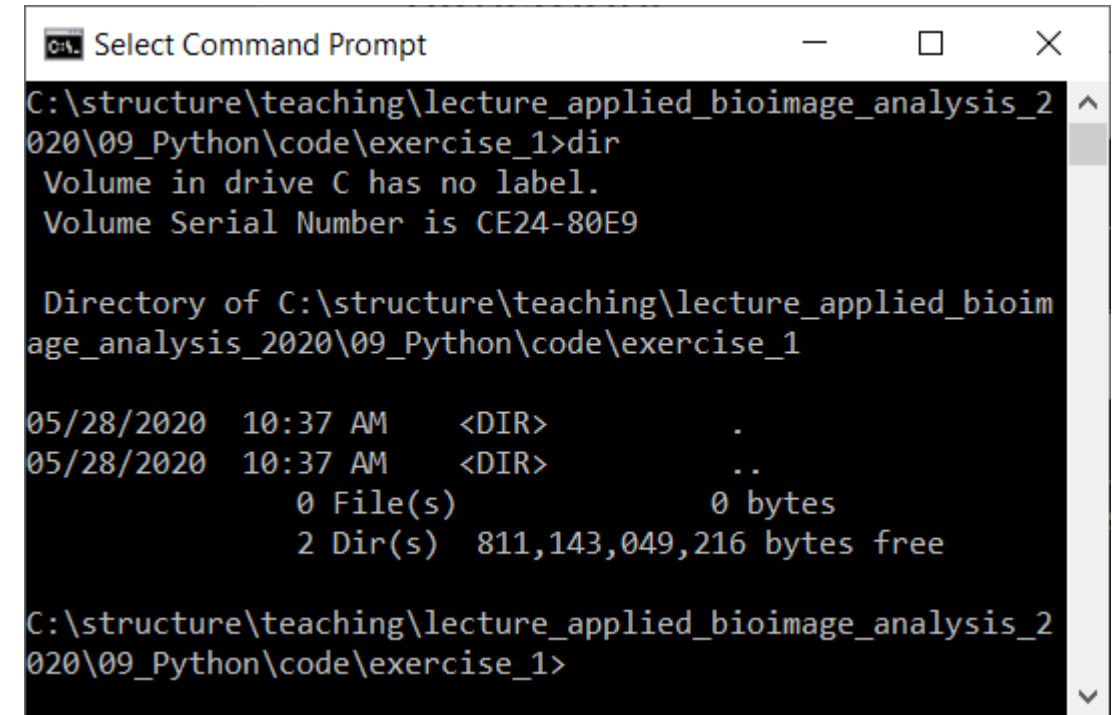
File names

- A.k.a. the Terminal or Eingabeaufforderung: Welcome to the 20<sup>th</sup> century!
- The `mkdir` command creates new directories.



```
Command Prompt
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python\code>mkdir exercise_1

Command Prompt
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python\code>cd exercise_1
```



```
Select Command Prompt
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python\code\exercise_1>dir
Volume in drive C has no label.
Volume Serial Number is CE24-80E9

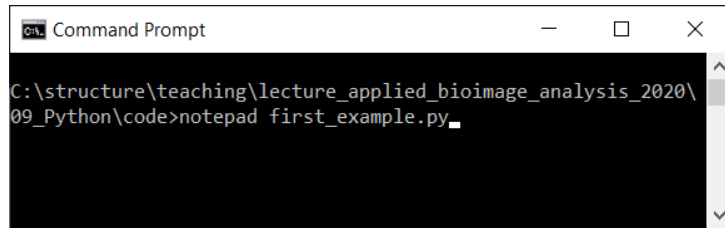
Directory of C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python\code\exercise_1

05/28/2020  10:37 AM    <DIR>          .
05/28/2020  10:37 AM    <DIR>          ..
               0 File(s)                0 bytes
               2 Dir(s)  811,143,049,216 bytes free

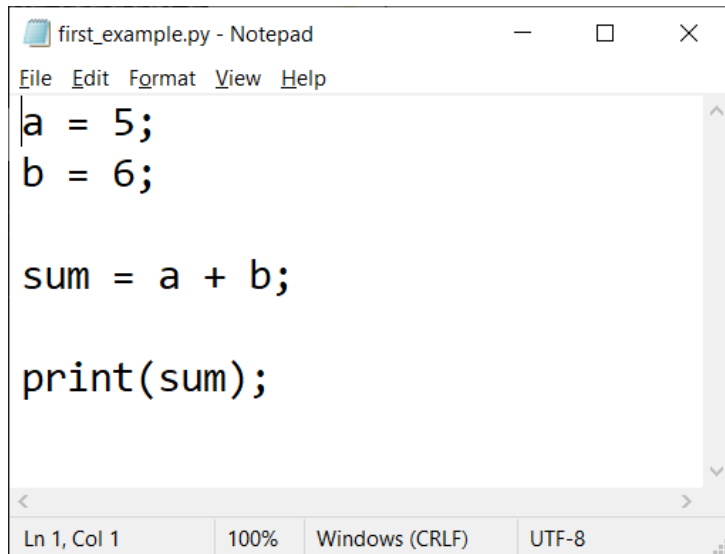
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python\code\exercise_1>
```

- Windows specific
- Notepad text editor

`notepad <filename>`



```
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python\code>notepad first_example.py
```



```
first_example.py - Notepad
File Edit Format View Help
a = 5;
b = 6;

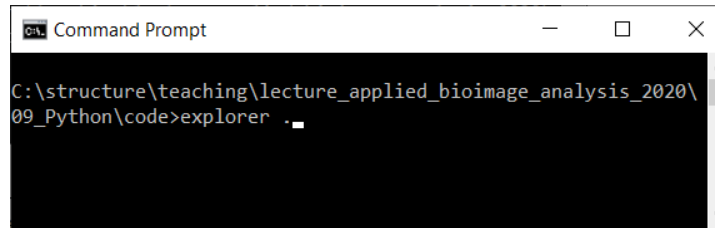
sum = a + b;

print(sum);

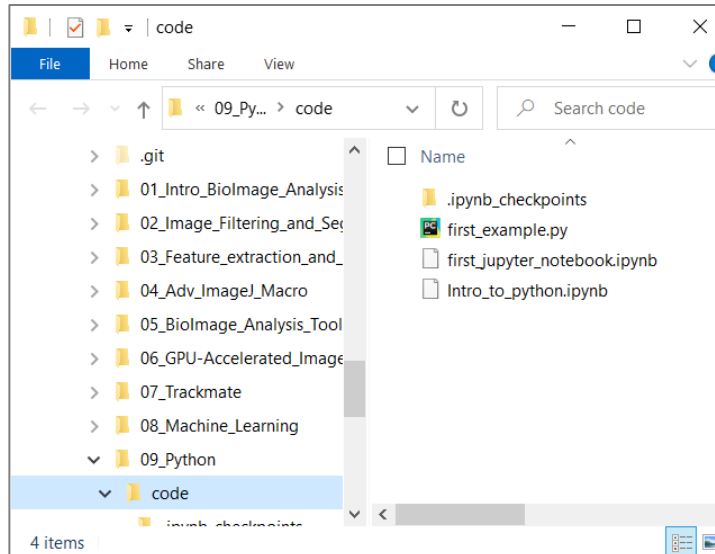
Ln 1, Col 1    100%    Windows (CRLF)    UTF-8
```

- Windows Explorer

`explorer .`

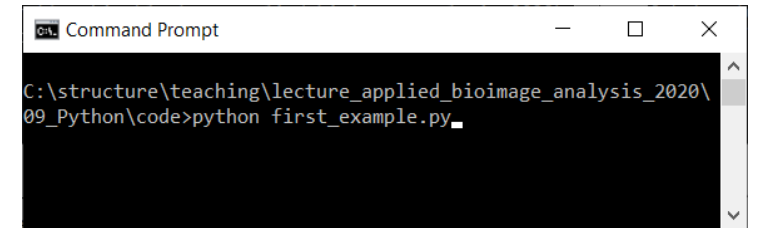


```
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python\code>explorer .
```

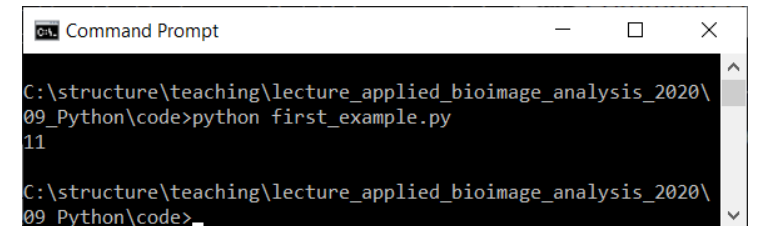


- Execute Python script

`python <filename>`



```
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python\code>python first_example.py
```



```
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python\code>python first_example.py
11
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python\code>
```

- Mac OS specific

- Text editor

`touch <filename>`

`open -e <filename>`

Create a  
new file

- Finder

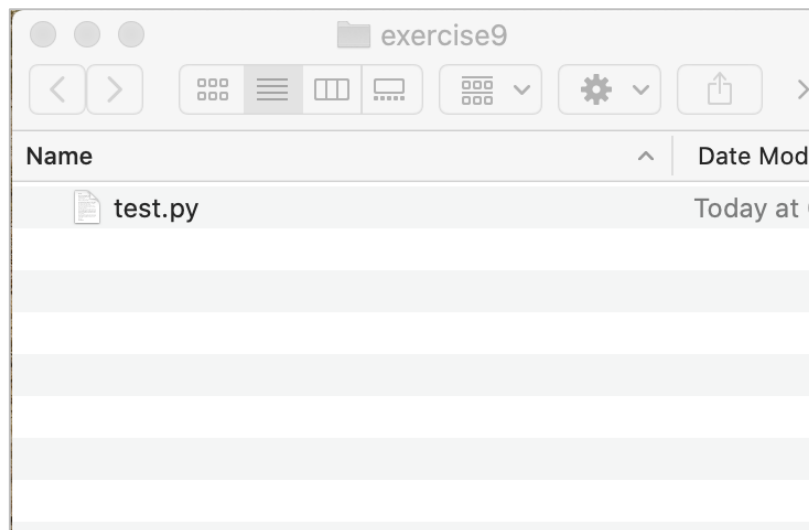
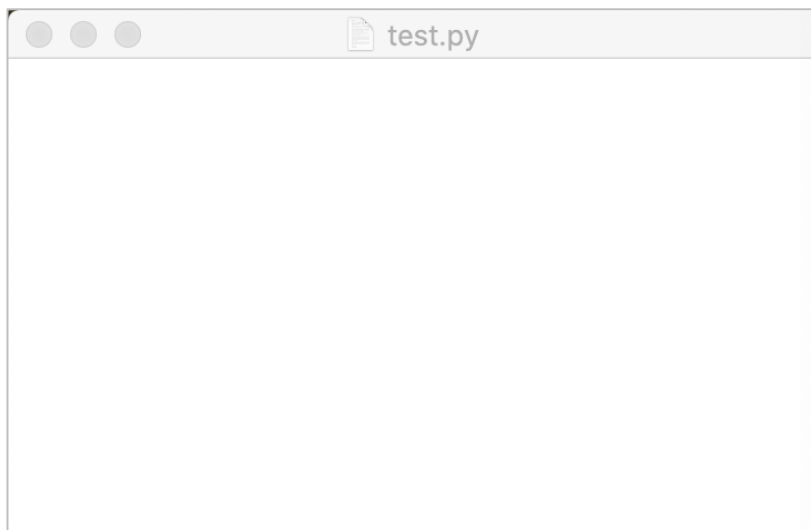
`open .`

- Execute Python script

`python <filename>`

```
exercise9 — -zsh — 51x24
haase@pcs-MacBook exercise9 % touch test.py
haase@pcs-MacBook exercise9 % open -e test.py
haase@pcs-MacBook exercise9 %
```

```
exercise9 — -zsh — 51x24
[haase@pcs-MacBook exercise9 % open .
```

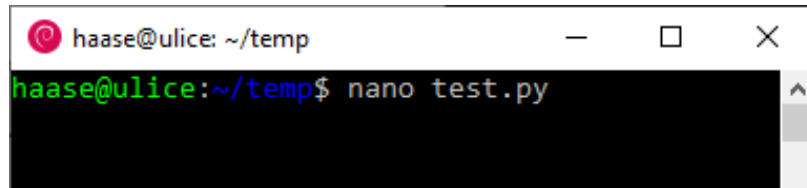


# The command line

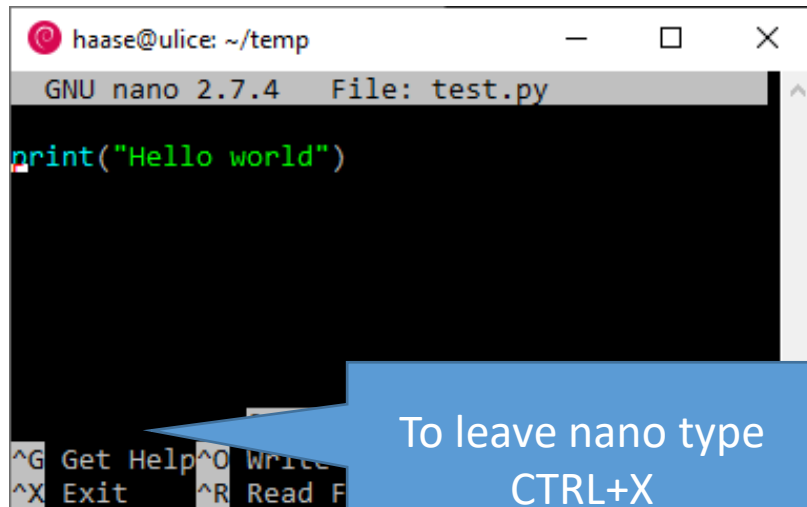
- Linux specific

- Nano text editor

`nano <filename>`



```
haase@ulice: ~/temp
haase@ulice:~/temp$ nano test.py
```

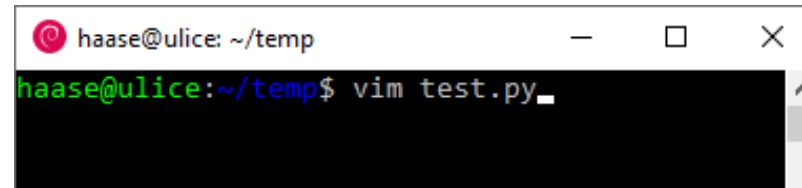


```
GNU nano 2.7.4 File: test.py
print("Hello world")
^G Get Help ^O Write Out
^X Exit ^R Read From Disk
```

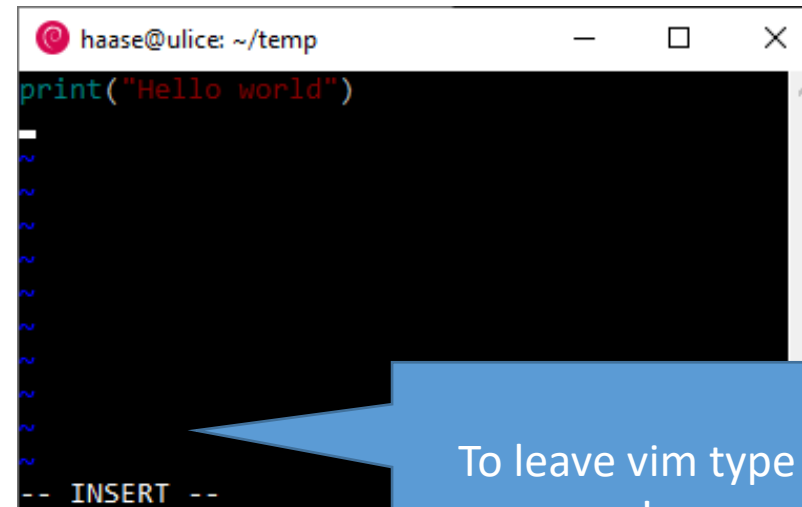
To leave nano type  
CTRL+X  
N

- vim

`vim <filename>`



```
haase@ulice: ~/temp
haase@ulice:~/temp$ vim test.py
```

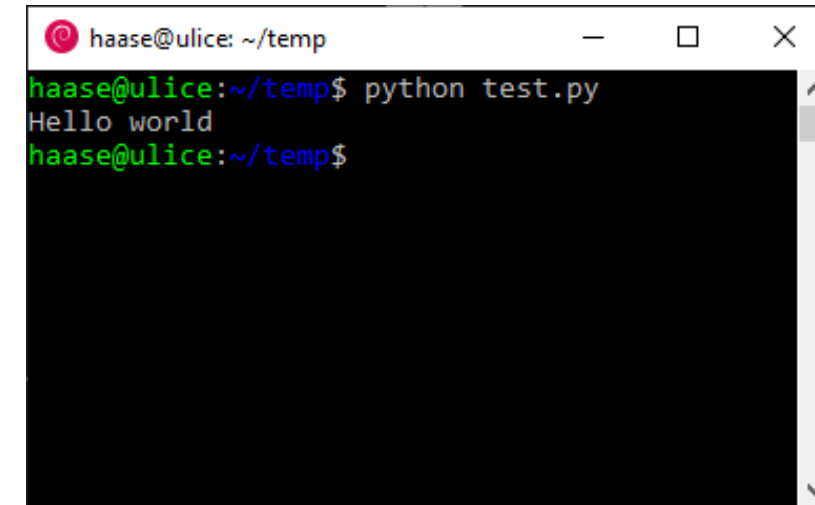


```
print("Hello world")
~
~
~
~
-- INSERT --
```

To leave vim type  
:q!

- Execute Python script

`python <filename>`



```
haase@ulice: ~/temp
haase@ulice:~/temp$ python test.py
Hello world
haase@ulice:~/temp$
```



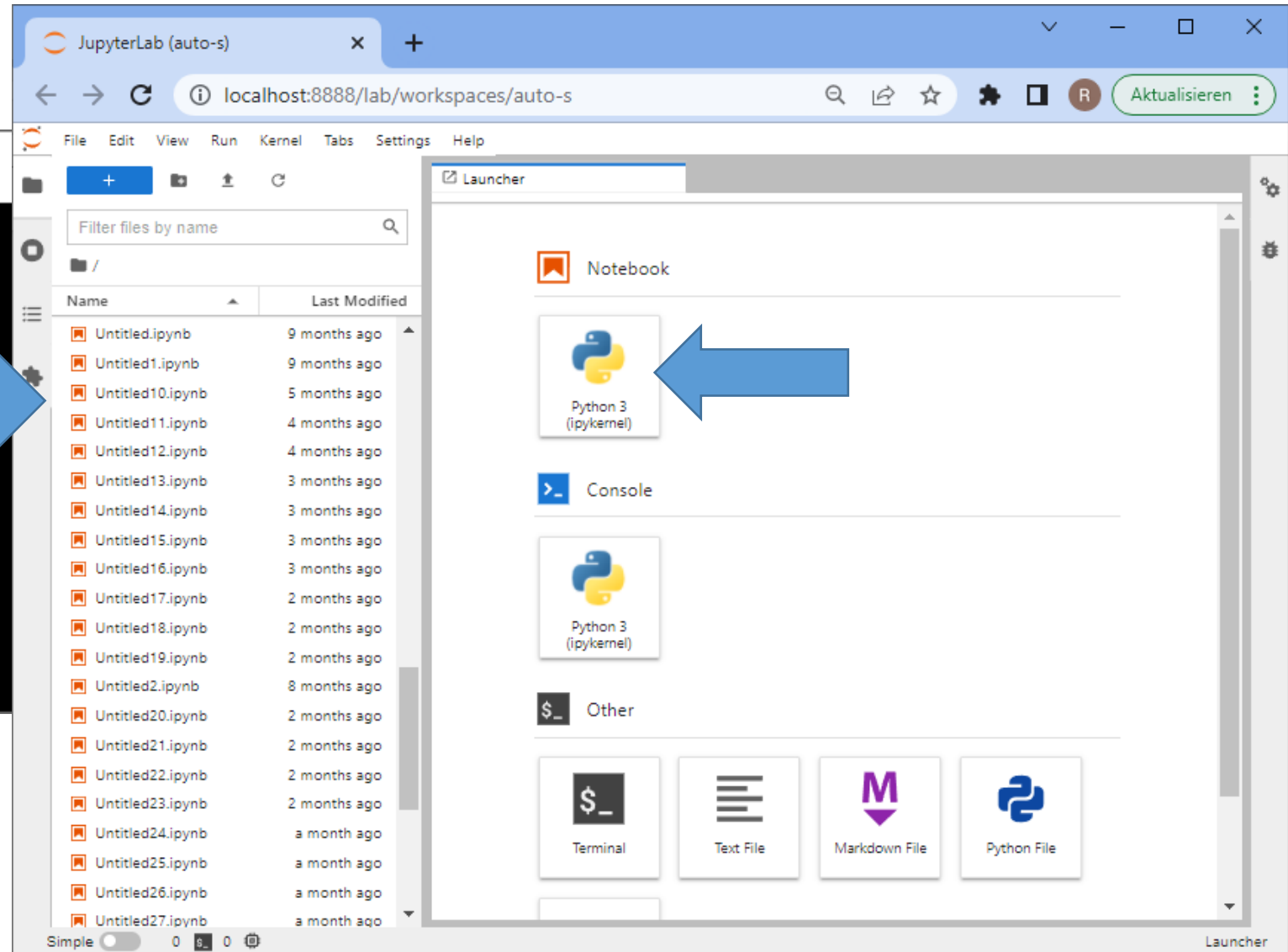
Robert Haase

October 2022

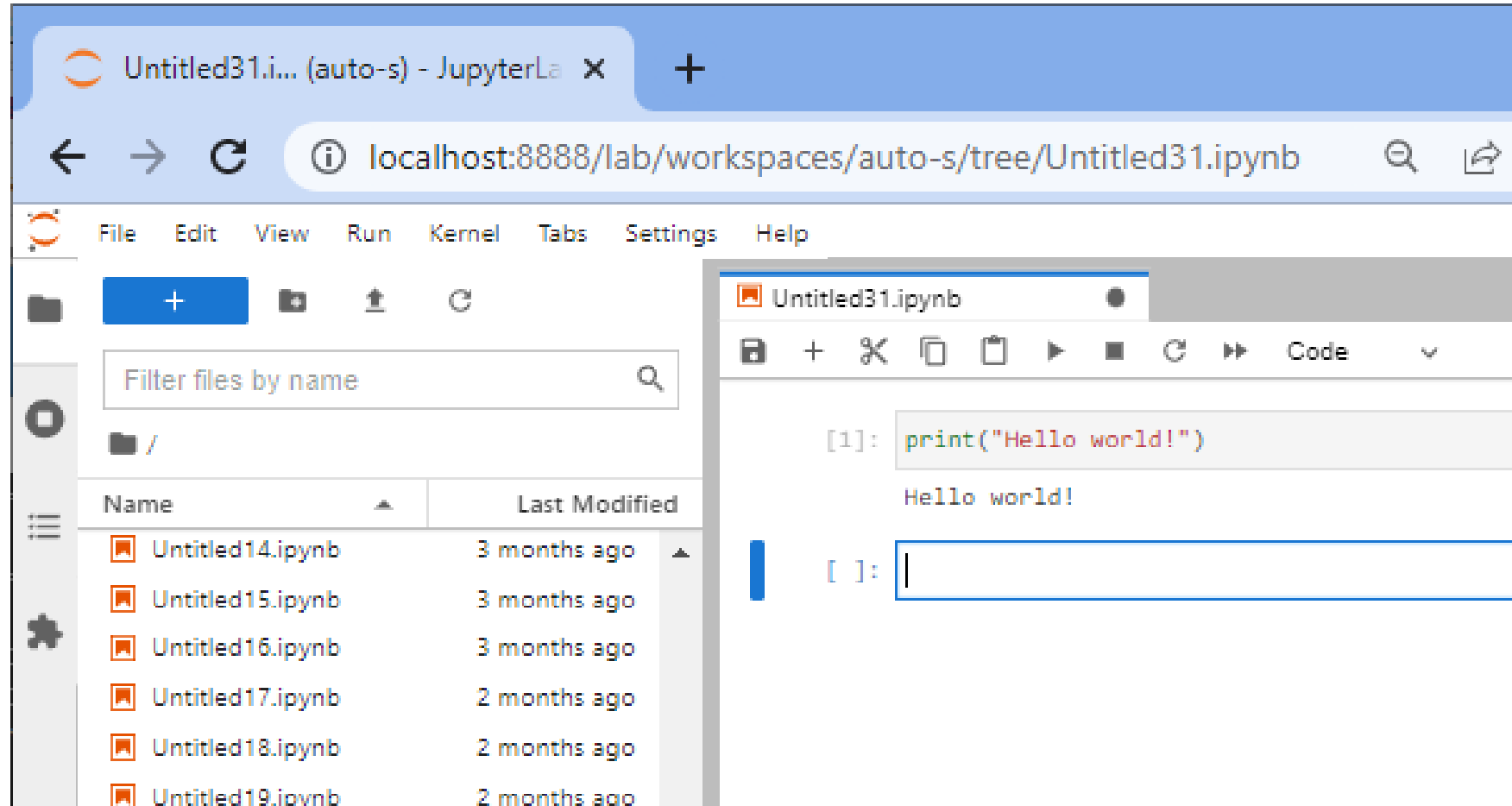


- Our programming environment for this semester

```
C:\Users\rober>conda deactivate - cond...  
  
c:\Users\rober>conda activate bio_39  
(bio_39) c:\Users\rober>jupyter lab
```



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

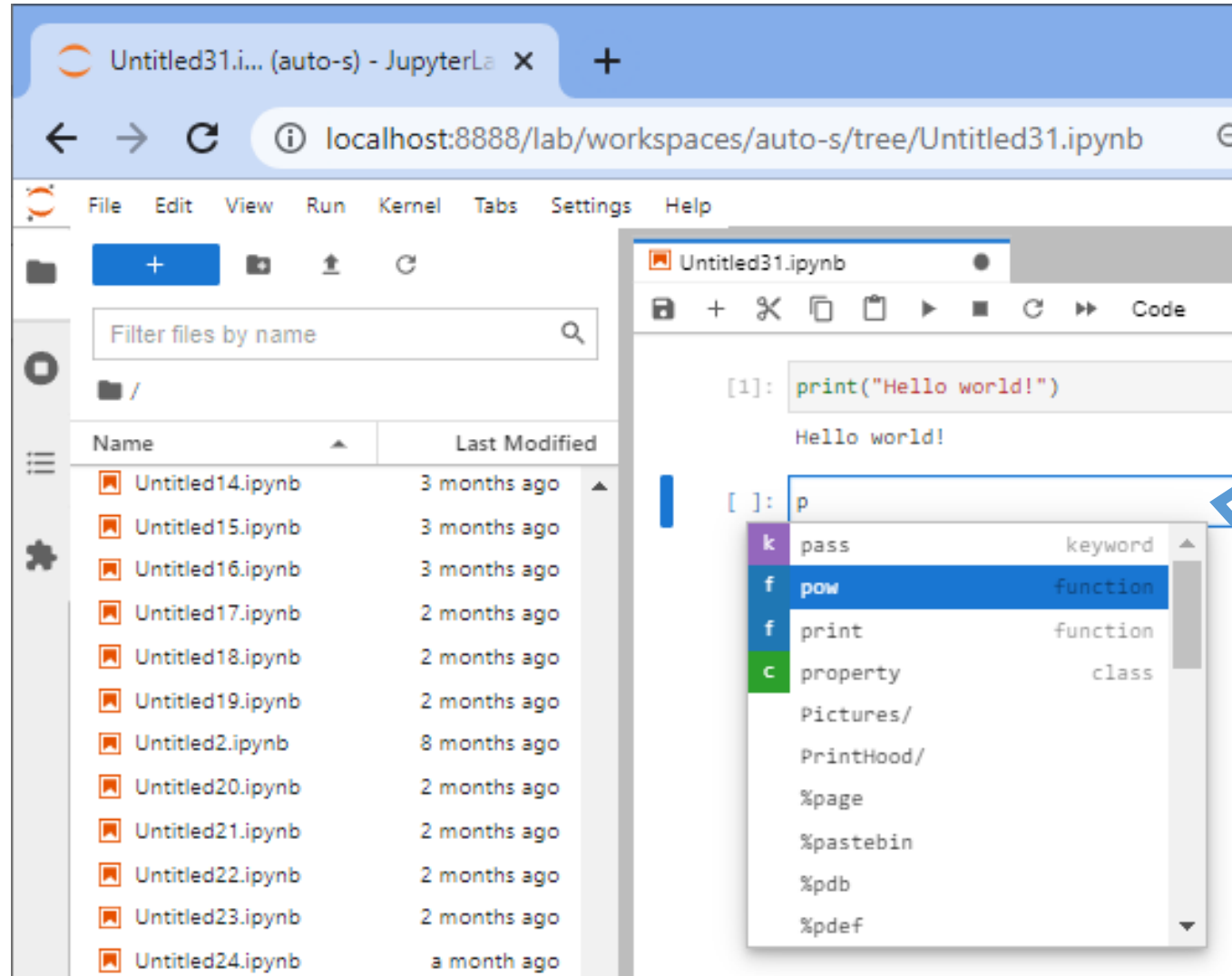


The screenshot shows the JupyterLab web interface. The top bar displays the browser tab 'Untitled31.i... (auto-s) - JupyterLa' and the URL 'localhost:8888/lab/workspaces/auto-s/tree/Untitled31.ipynb'. The left sidebar contains a file browser with a search bar and a list of files: 'Untitled14.ipynb', 'Untitled15.ipynb', 'Untitled16.ipynb', 'Untitled17.ipynb', 'Untitled18.ipynb', and 'Untitled19.ipynb'. The main area shows a code editor for 'Untitled31.ipynb' with a toolbar and a code cell containing the following code and output:

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

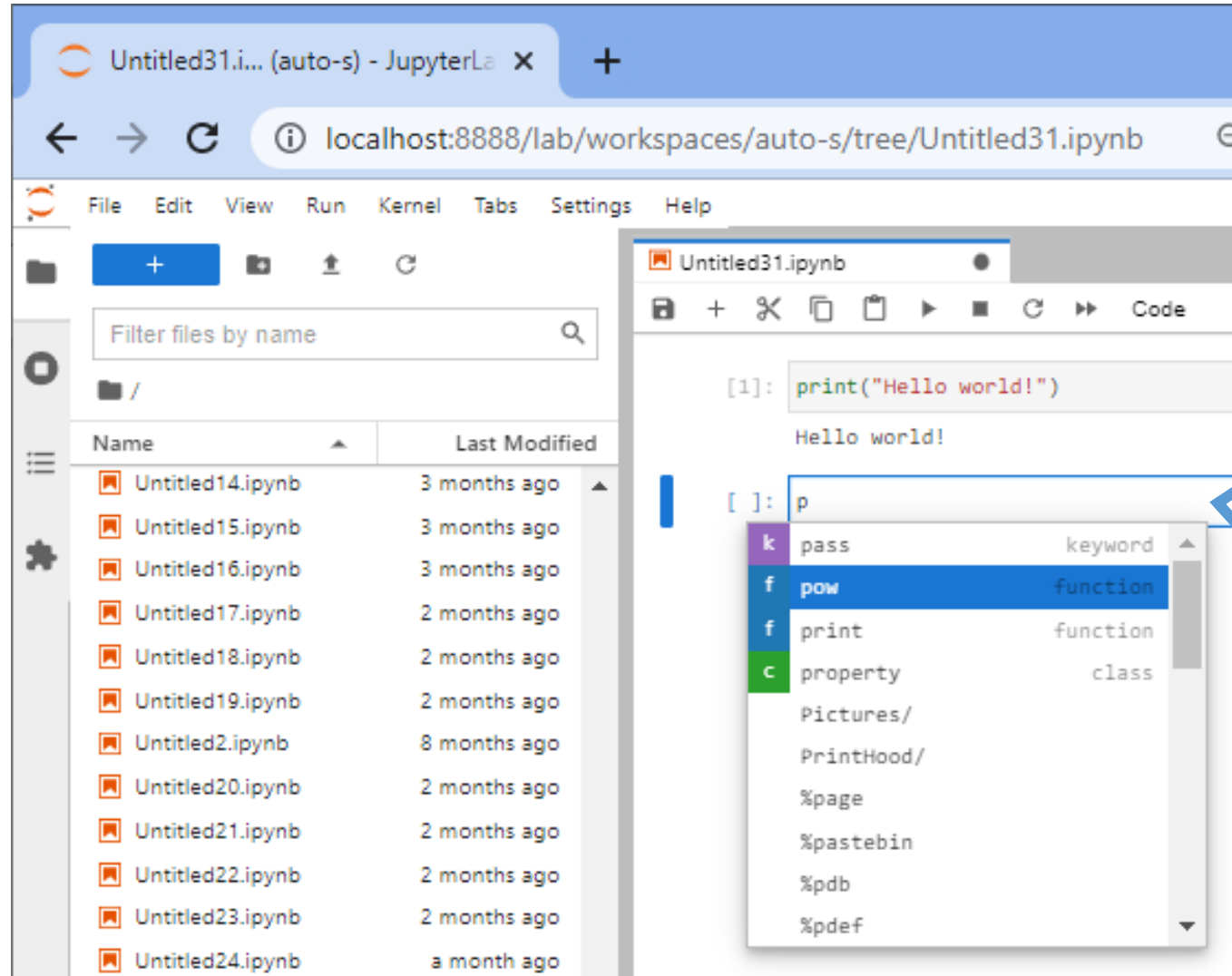
A blue box highlights the code cell, and a callout bubble points to it with the text 'SHIFT + ENTER to execute a code cell'.

- Context-specific help, auto-completion



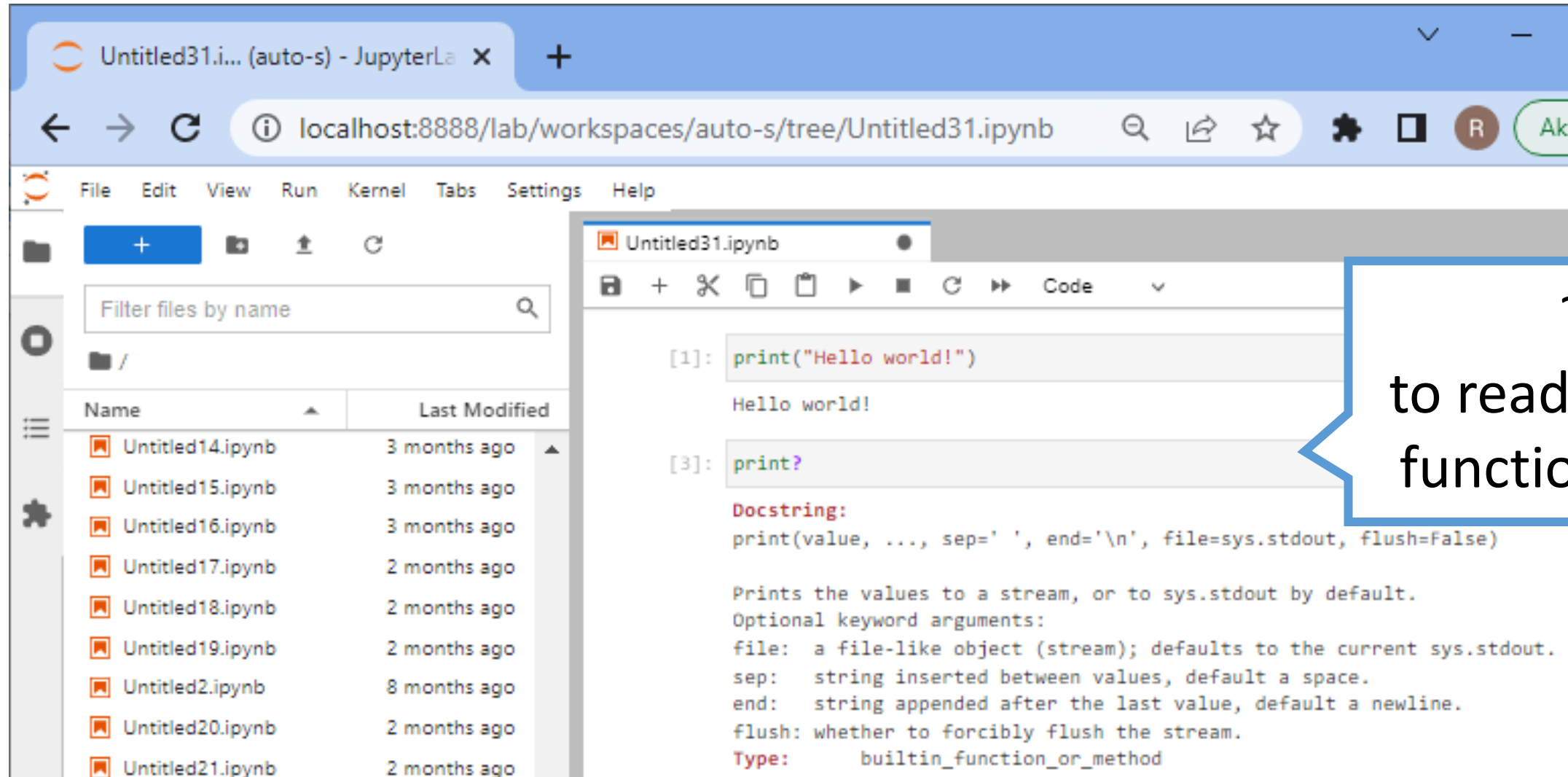
TAB  
to open auto-  
completion

- 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 displays a Jupyter cell. The cell contains the code `print("Hello world!")` and its output `Hello world!`. Below this, the same cell shows the code `print?` and its docstring. A blue callout box points to the docstring with the text `? to read what a function does`.

Name	Last Modified
Untitled14.ipynb	3 months ago
Untitled15.ipynb	3 months ago
Untitled16.ipynb	3 months ago
Untitled17.ipynb	2 months ago
Untitled18.ipynb	2 months ago
Untitled19.ipynb	2 months ago
Untitled2.ipynb	8 months ago
Untitled20.ipynb	2 months ago
Untitled21.ipynb	2 months ago

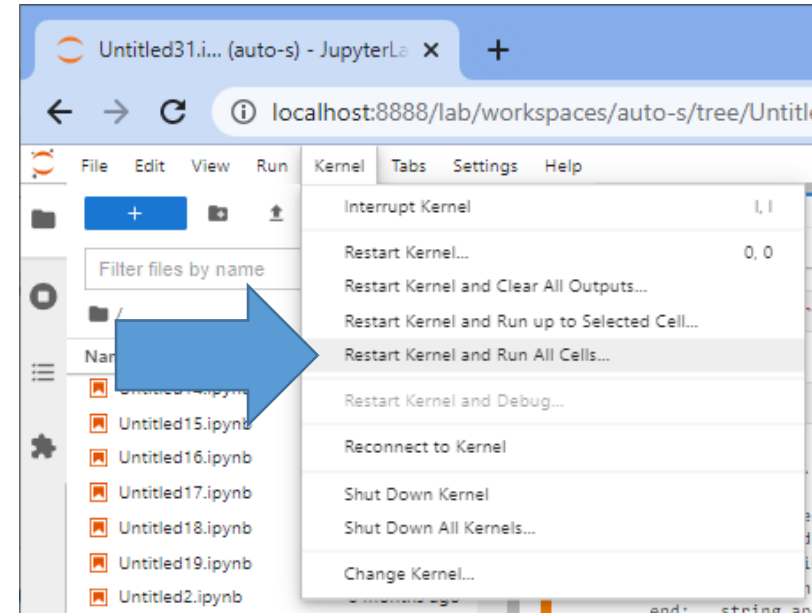
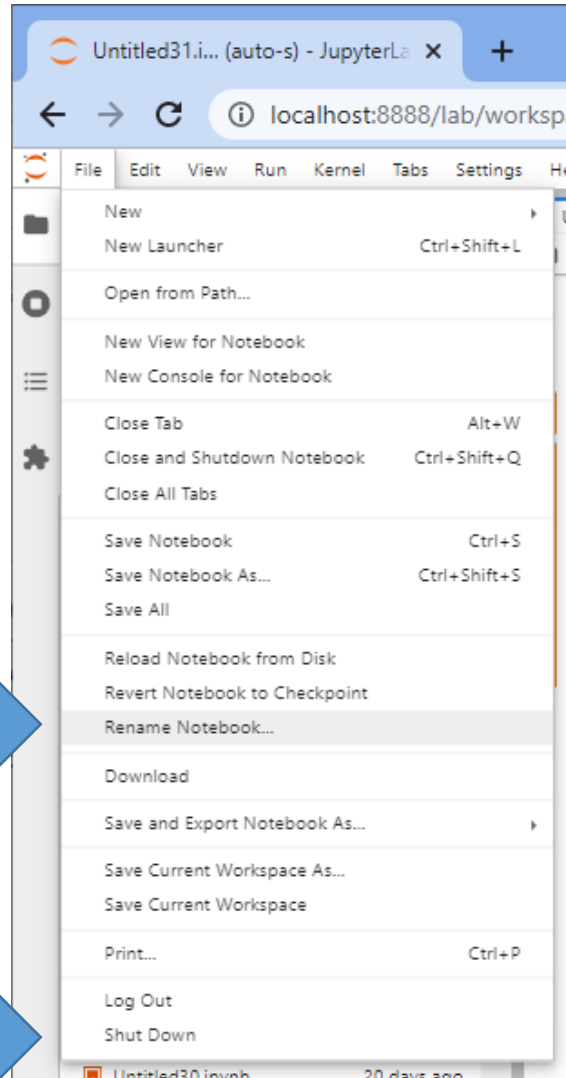
```
[1]: print("Hello world!")
Hello world!

[3]: print?

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
```

- Saving / renaming / closing



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

# Python programming basics

Robert Haase

October 2022

- 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

- 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()`