Python Essentials. Lection 1

Python Essentials. Technical features.

- General purpose programming language
- Strongly typed
- Dynamic
- Interpreted
- Multiparadigm
- CPyton implementation mostly used

Versions / Timeline

Python 1.0
January 1994

- Python 1.5 - December 31, 1997
- Python 1.6 - September 5, 2000

Python 2.0
October 16, 2000

- Python 2.1 - April 17, 2001
- Python 2.2 - December 21, 2001
- Python 2.3 - July 29, 2003
- Python 2.4 - November 30, 2004
- Python 2.5 - September 19, 2006
- Python 2.6 - October 1, 2008
- Python 2.7 - July 3, 2010

Python 3.0
December 3, 2008

- Python 3.1 - June 27, 2009
- Python 3.2 - February 20, 2011
- Python 3.3 - September 29, 2012
- Python 3.4 - March 16, 2014
- Python 3.5 - September 13, 2015
- Python 3.6 - December 23, 2016
- Python 3.7 - June 27, 2018
- Python 3.8 - October 14, 2019
- Python 3.9 – October 05, 2020
- Python 3.10 – October 04, 2021
- Python 3.11 – October 03, 2022

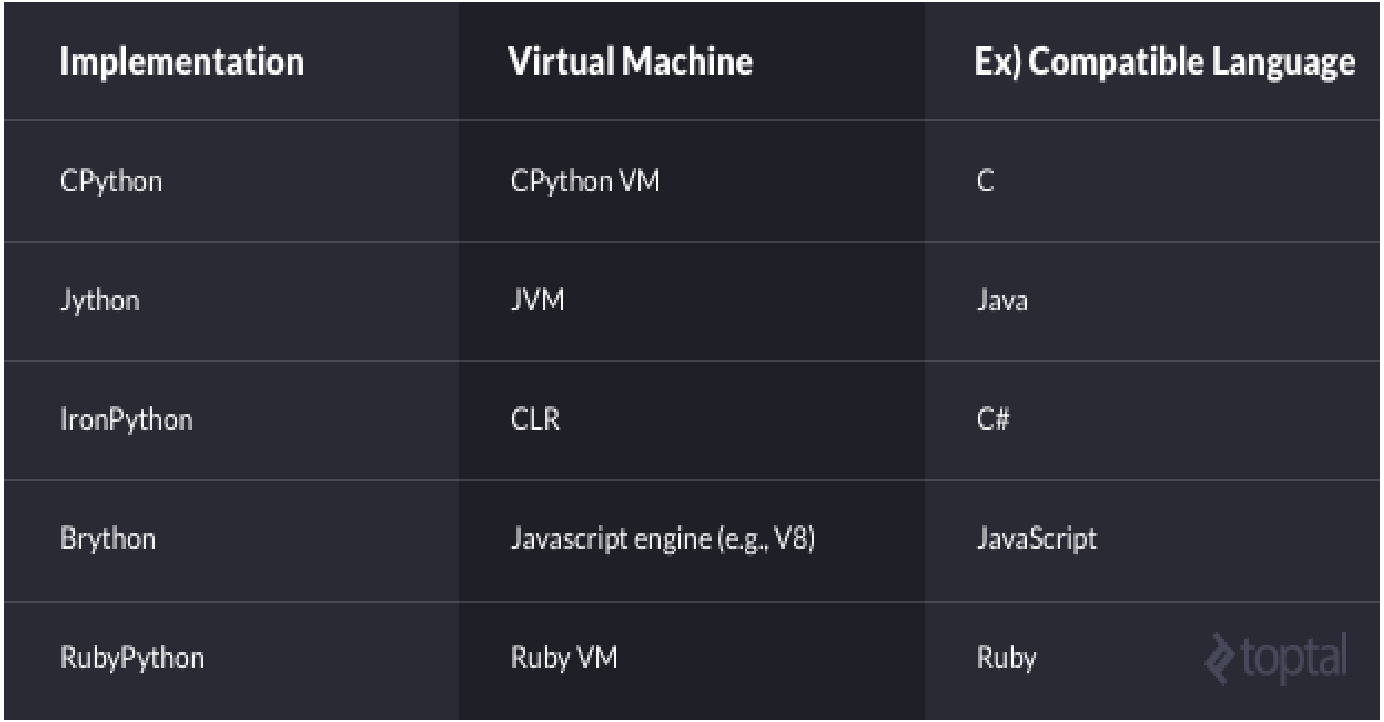
Python Essentials. Why Python?

- Automation of simple routine tasks
- Scientific applications
- Desktop applications
- Android
- Web applications (Django, Flask)
- Machine learning (Tenser Flow library)
- Simple (elegant) syntax
- Lots of free libraries
- Easy to learn
- Popular in the IT industry

Python Essentials. Python 2 vs Python 3.

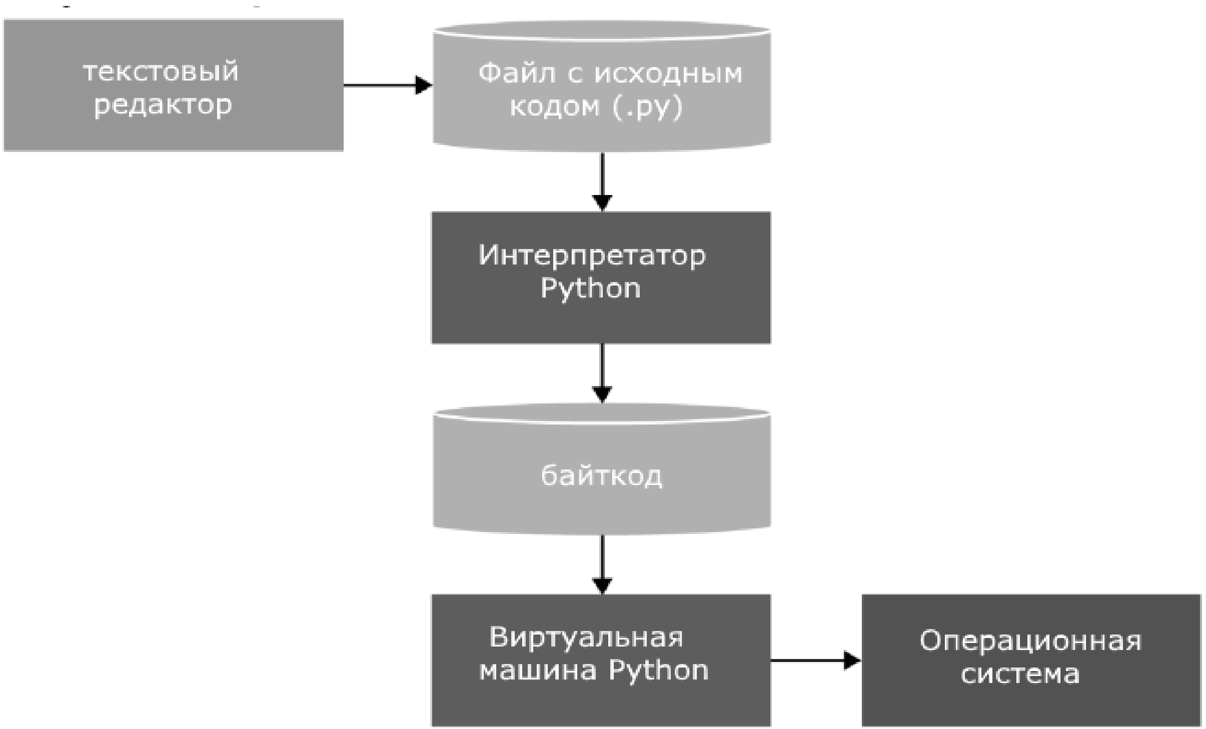
- Python 3 today and future
- Python 2 support will finish in 2020 (this year)
- Python 3 and Python 2 partially uncompatible
- Python 3 use Unicode by default
- Mostly libraries rebuilt for Python 3
- Ease of migration from Python 3 to Python 2

Type Python Implementation. Virtual Machine



Implementation	Virtual Machine	Ex) Compatible Language
CPython	CPython VM	C
Jython	JVM	Java
IronPython	CLR	C#
Brython	Javascript engine (e.g., V8)	JavaScript
RubyPython	Ruby VM	Ruby

Execute of Python code



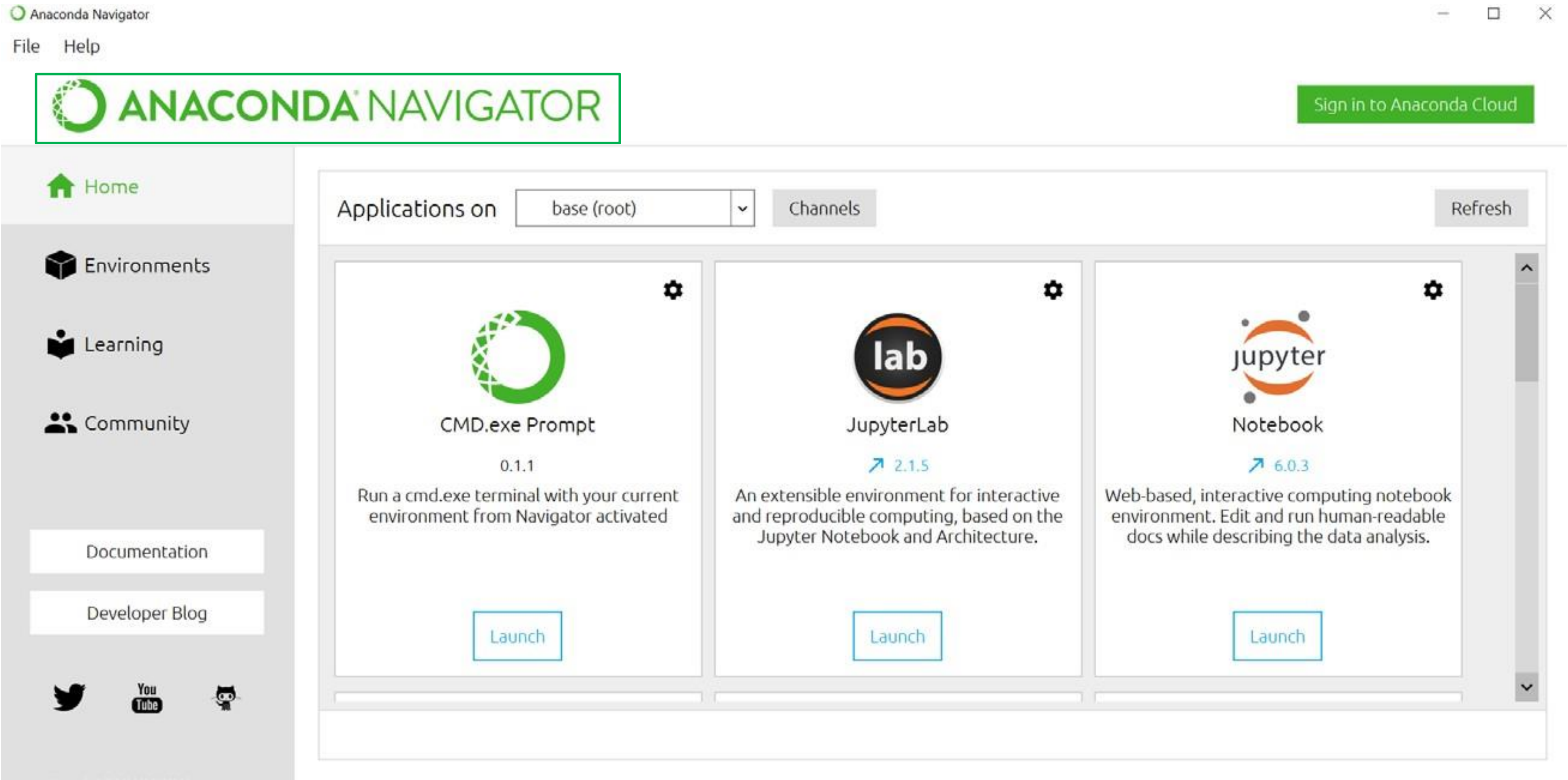
Python Essentials. Tools for coding

- General purpose editors (less fuctional [Sublime, Visual Code, Notepad++])
- IDEs (huge code size [PyCharm, Spyder])
- Notebook shells

Python Essentials. Notebook shells

- Markdown and visualization support
- Ease to make documented code
- Good usability with REPL mode (read-evaluate-print loop)
- Pretty good for learning programming language
- Mostly popular is Jupiter Notebook (.ipynb - extentions)

Python Essentials. Using Jupiter Notebook with Anaconda



CMD.exe Prompt

0.1.1

Run a cmd.exe terminal with your current environment from Navigator activated

JupyterLab

2.1.5

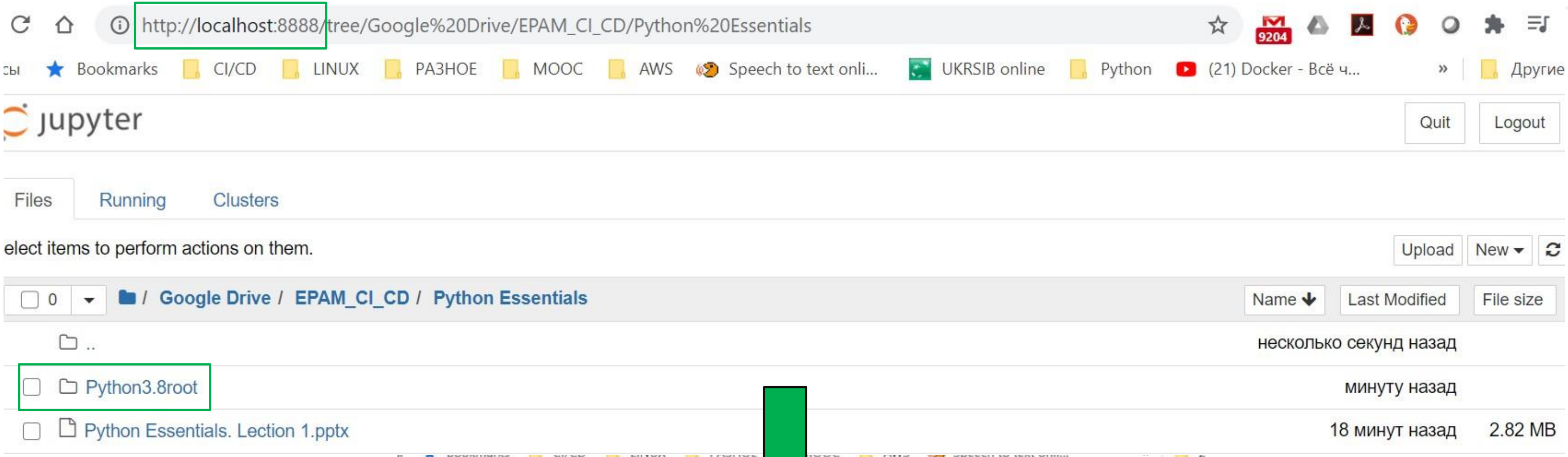
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Notebook

6.0.3

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Python Essentials. Preparation for using Jupiter Notebook

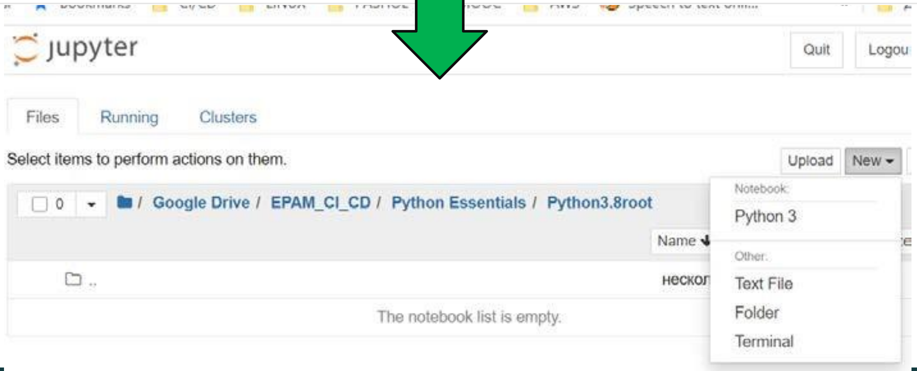


http://localhost:8888/tree/Google%20Drive/EPAM_CI_CD/Python%20Essentials

elect items to perform actions on them.

□ 0 / Google Drive / EPAM_CI_CD / Python Essentials

..	несколько секунд назад	
□ Python3.8root	минуту назад	
□ Python Essentials. Lection 1.pptx	18 минут назад	2.82 MB



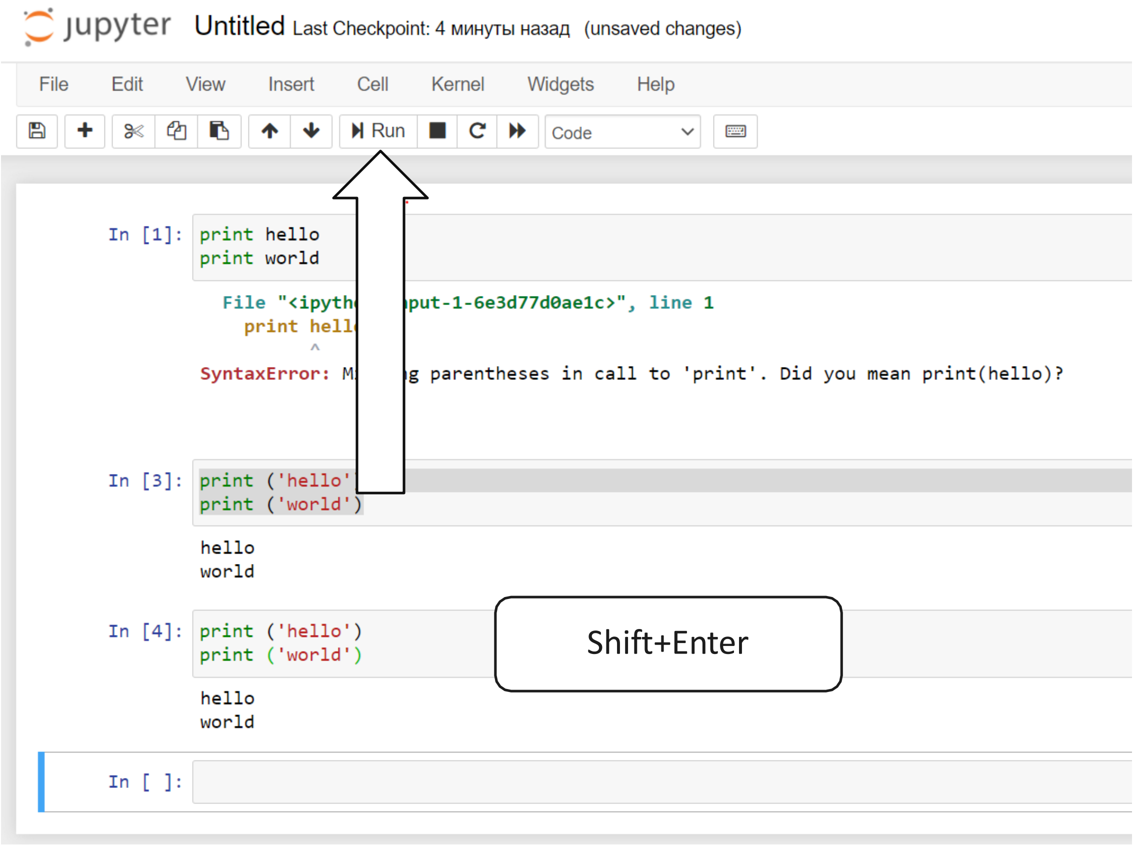
Select items to perform actions on them.

□ 0 / Google Drive / EPAM_CI_CD / Python Essentials / Python3.8root

.. несколі

The notebook list is empty.

Python Essentials. Another one “Hello World”



```
In [1]: print hello
        print world

          File "<ipyth    put-1-6e3d77d0ae1c>", line 1
            print hell
                      ^
        SyntaxError: M    g parentheses in call to 'print'. Did you mean print(hello)?
```

```
In [3]: print ('hello'
        print ('world')

        hello
        world
```

```
In [4]: print ('hello')
        print ('world')

        hello
        world
```

```
In [ ]:
```

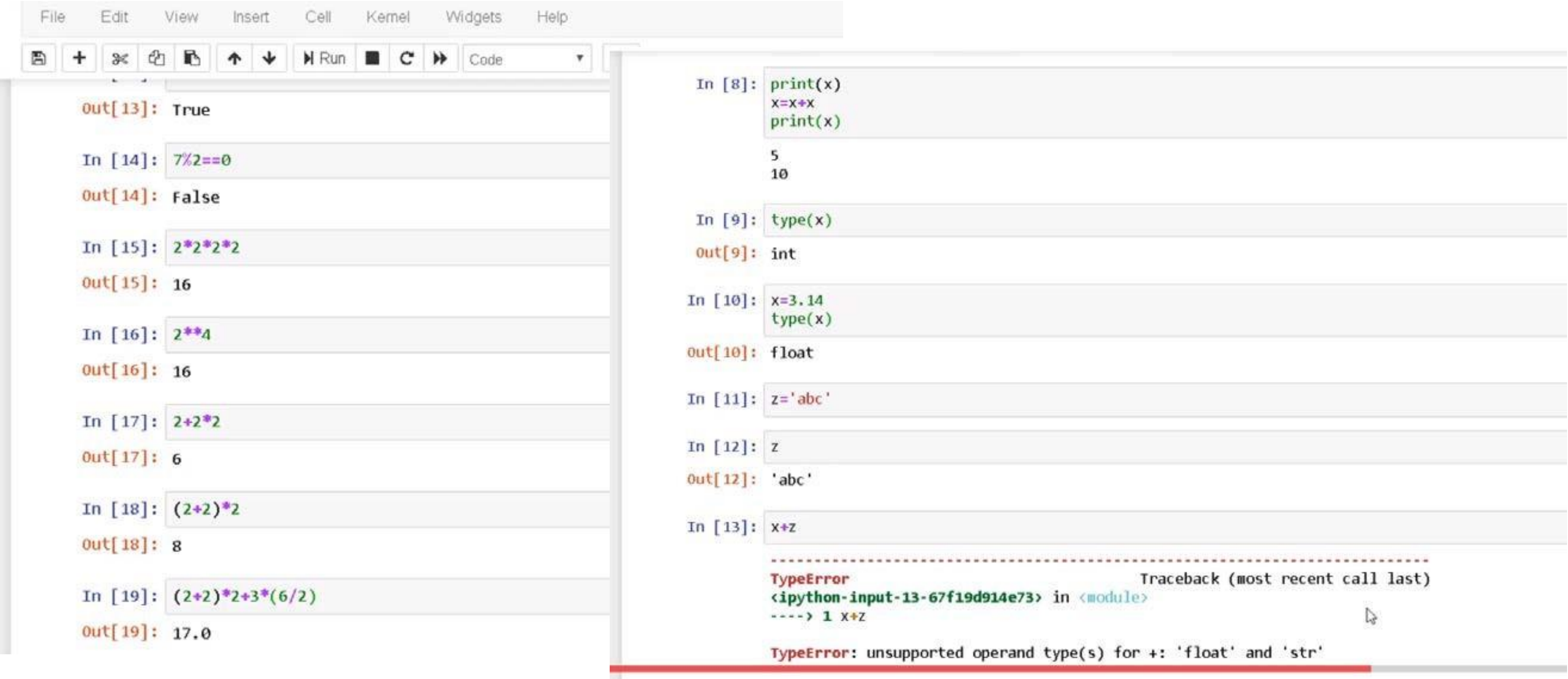
Shift+Enter

Python Essentials. Frequently used data types.

Name	Type	Description
Integer	int	1, 2, 3
Floating	float	1.1, 2.67887, 5.0
String	str	“Hello World”, ‘Hello 2020 Fall’
List	list	[1, 2.0, “Good Morning”]
Dictionarie	dict	{“vasya” : “+380653333331”, “masya” : “+380654441113”}
Tuple	tuple	(1, 2.0, “Freeze!”)
Set	set	{“1”, “2”, “3”}
Boolean	bool	Boolean values: True or False

MORE: https://www.w3schools.com/python/python_datatypes.asp

Python Essentials. Numbers and variables examples.

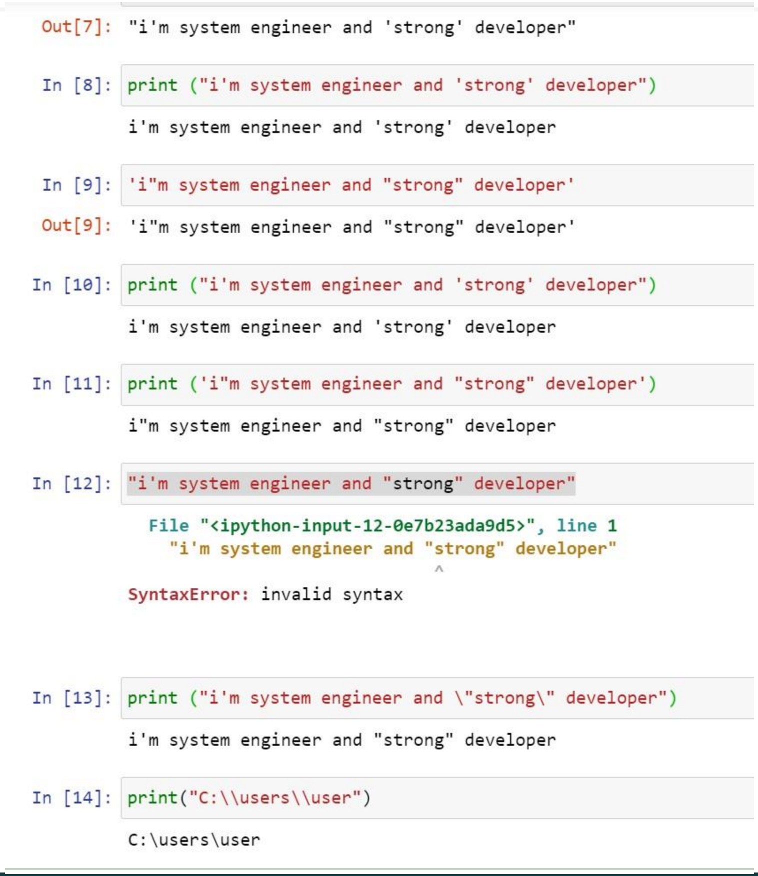


```
Out[13]: True

In [14]: 7%2==0
Out[14]: False

In [15]: 2*2*2*2
Out[15]: 16

In [16]: 2**4
Out[16]: 16

In [17]: 2+2*2
Out[17]: 6

In [18]: (2+2)*2
Out[18]: 8

In [19]: (2+2)*2+3*(6/2)
Out[19]: 17.0
```

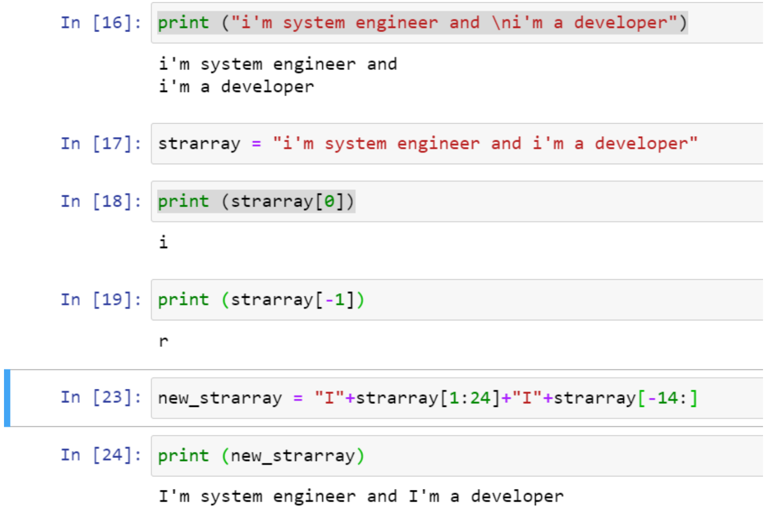
```
In [8]:  print(x)
         x=x+x
         print(x)

         5
         10

In [9]:  type(x)
Out[9]:  int

In [10]: x=3.14
         type(x)
Out[10]: float

In [11]: z='abc'

In [12]: z
Out[12]: 'abc'

In [13]: x+z

         ---------------------------------------------------------------------------
         TypeError                                 Traceback (most recent call last)
         <ipython-input-13-67f19d914e73> in <module>
         ----> 1 x+z

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

MORE: https://www.w3schools.com/python/python_datatypes.asp

Python Essentials. String examples.

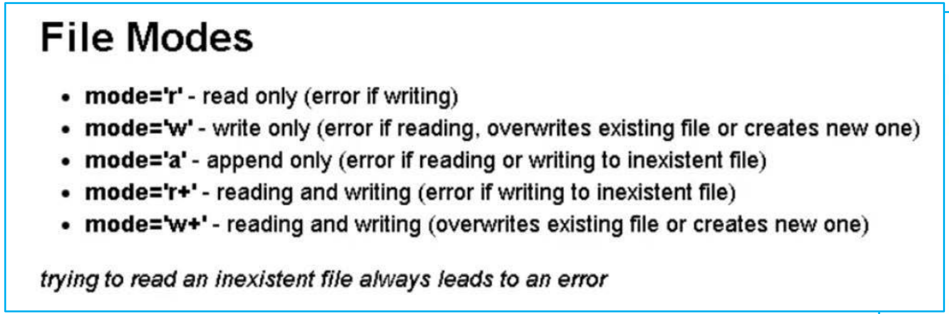


```
Out[7]: "i'm system engineer and 'strong' developer"

In [8]: print ("i'm system engineer and 'strong' developer")

        i'm system engineer and 'strong' developer

In [9]: 'i"m system engineer and "strong" developer'

Out[9]: 'i"m system engineer and "strong" developer'

In [10]: print ("i'm system engineer and 'strong' developer")

         i'm system engineer and 'strong' developer

In [11]: print ('i"m system engineer and "strong" developer')

         i"m system engineer and "strong" developer

In [12]: "i'm system engineer and "strong" developer"

           File "<ipython-input-12-0e7b23ada9d5>", line 1
             "i'm system engineer and "strong" developer"
                                            ^
         SyntaxError: invalid syntax

In [13]: print ("i'm system engineer and \"strong\" developer")

         i'm system engineer and "strong" developer

In [14]: print("C:\\users\\user")

         C:\users\user
```



```
In [16]: print ("i'm system engineer and \ni'm a developer")

         i'm system engineer and
         i'm a developer

In [17]: strarray = "i'm system engineer and i'm a developer"

In [18]: print (strarray[0])

         i

In [19]: print (strarray[-1])

         r

In [23]: new_strarray = "I"+strarray[1:24]+"I"+strarray[-14:]

In [24]: print (new_strarray)

         I'm system engineer and I'm a developer
```

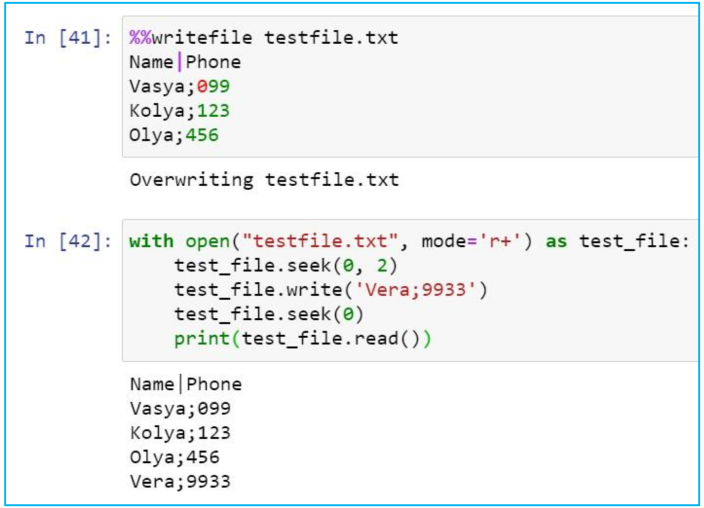
Python Essentials. Operating with files examples.



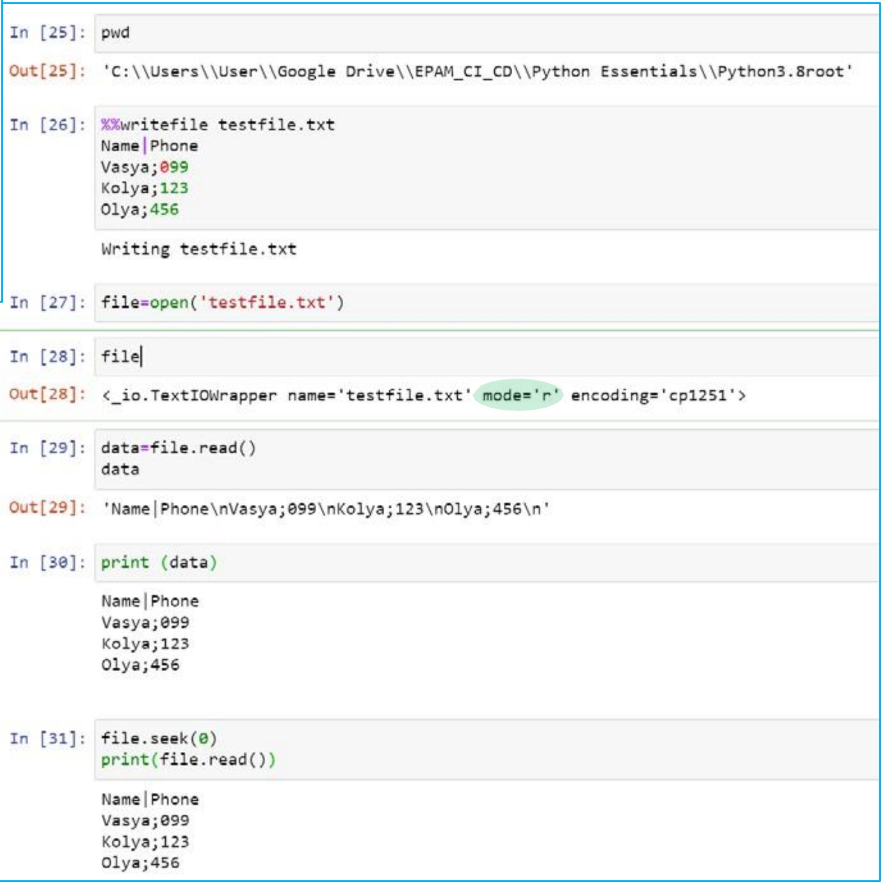
File Modes

- **mode='r'** - read only (error if writing)
- **mode='w'** - write only (error if reading, overwrites existing file or creates new one)
- **mode='a'** - append only (error if reading or writing to inexistent file)
- **mode='r+'** - reading and writing (error if writing to inexistent file)
- **mode='w+'** - reading and writing (overwrites existing file or creates new one)

trying to read an inexistent file always leads to an error

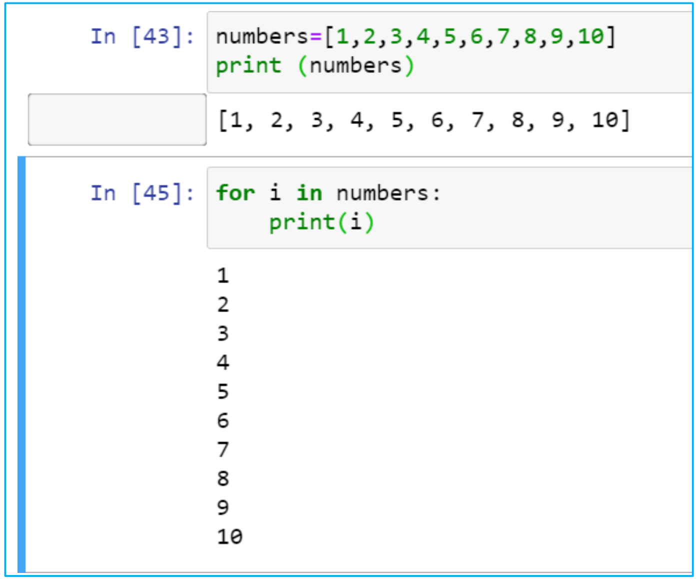


```
In [41]: %%writefile testfile.txt
         Name|Phone
         Vasya;099
         Kolya;123
         Olya;456

         Overwriting testfile.txt

In [42]: with open("testfile.txt", mode='r+') as test_file:
             test_file.seek(0, 2)
             test_file.write('Vera;9933')
             test_file.seek(0)
             print(test_file.read())

         Name|Phone
         Vasya;099
         Kolya;123
         Olya;456
         Vera;9933
```



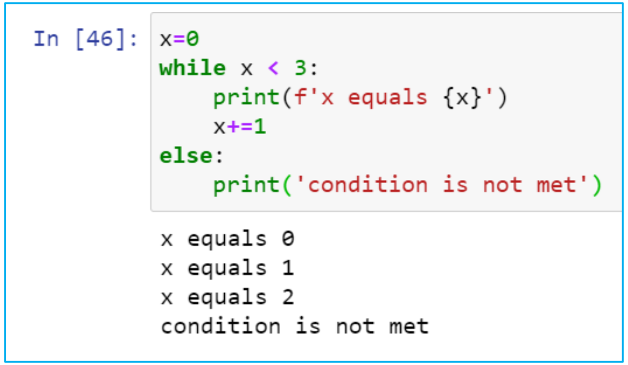
```
In [25]: pwd

Out[25]: 'C:\\Users\\User\\Google Drive\\EPAM_CI_CD\\Python Essentials\\Python3.8root'

In [26]: %%writefile testfile.txt
         Name|Phone
         Vasya;099
         Kolya;123
         Olya;456

         Writing testfile.txt

In [27]: file=open('testfile.txt')

In [28]: file

Out[28]: <_io.TextIOWrapper name='testfile.txt' mode='r' encoding='cp1251'>

In [29]: data=file.read()
         data

Out[29]: 'Name|Phone\nVasya;099\nKolya;123\nOlya;456\n'

In [30]: print (data)

         Name|Phone
         Vasya;099
         Kolya;123
         Olya;456


In [31]: file.seek(0)
         print(file.read())

         Name|Phone
         Vasya;099
         Kolya;123
         Olya;456
```

Python Essentials. Loop examples.



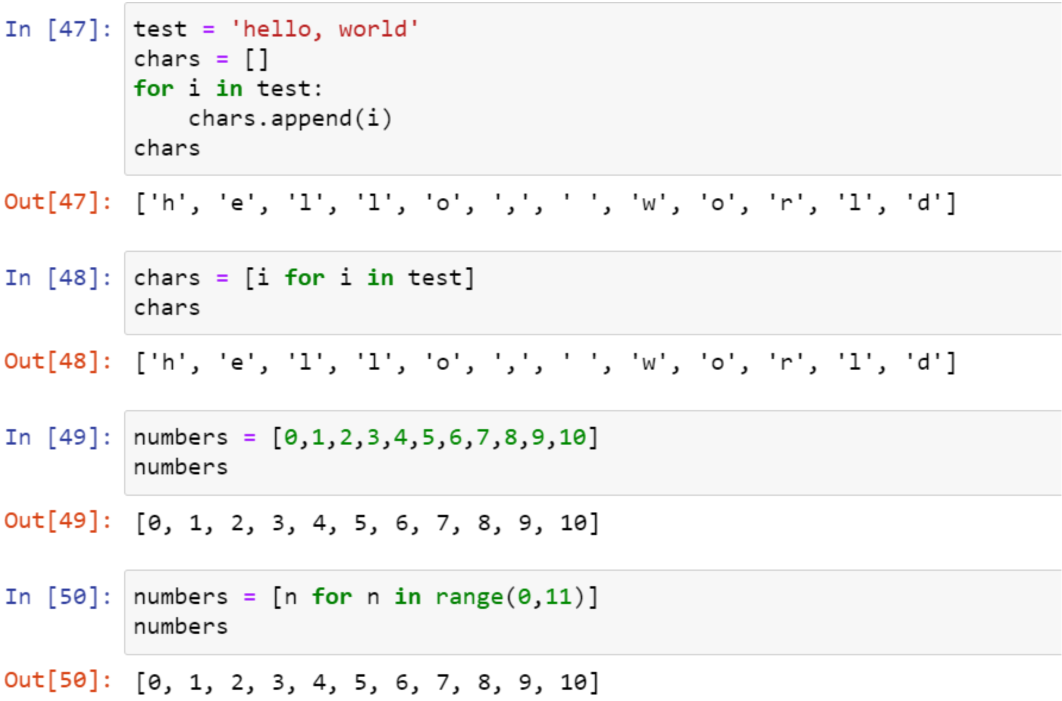
```
In [43]: numbers=[1,2,3,4,5,6,7,8,9,10]
         print (numbers)

         [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
In [45]: for i in numbers:
             print(i)

         1
         2
         3
         4
         5
         6
         7
         8
         9
         10
```



```
In [46]: x=0
         while x < 3:
             print(f'x equals {x}')
             x+=1
         else:
             print('condition is not met')

         x equals 0
         x equals 1
         x equals 2
         condition is not met
```

Python Essentials. Operating with list examples.



```
In [47]: test = 'hello, world'
         chars = []
         for i in test:
             chars.append(i)
         chars

Out[47]: ['h', 'e', 'l', 'l', 'o', ',', ' ', 'w', 'o', 'r', 'l', 'd']

In [48]: chars = [i for i in test]
         chars

Out[48]: ['h', 'e', 'l', 'l', 'o', ',', ' ', 'w', 'o', 'r', 'l', 'd']

In [49]: numbers = [0,1,2,3,4,5,6,7,8,9,10]
         numbers

Out[49]: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

In [50]: numbers = [n for n in range(0,11)]
         numbers

Out[50]: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

Q&A

Thank you!