Coding seminar

Lesson 1: Hello, World!

Ikue Hirata

Contents

What is/why Python?

Jupyter: Coding interface

Syntax – grammars

Variables

Types – int, float, string, boolean

Arithmetic

Control flows/loops

Functions

What is/why Python?

An artificial language for coding

Easy to learn/write (compared to other languages)

Widely used in science/engineering
(AI, deep learning, etc.)

Many libraries (= set of tools for common processes)

Free!

Python libraries

Mathematics

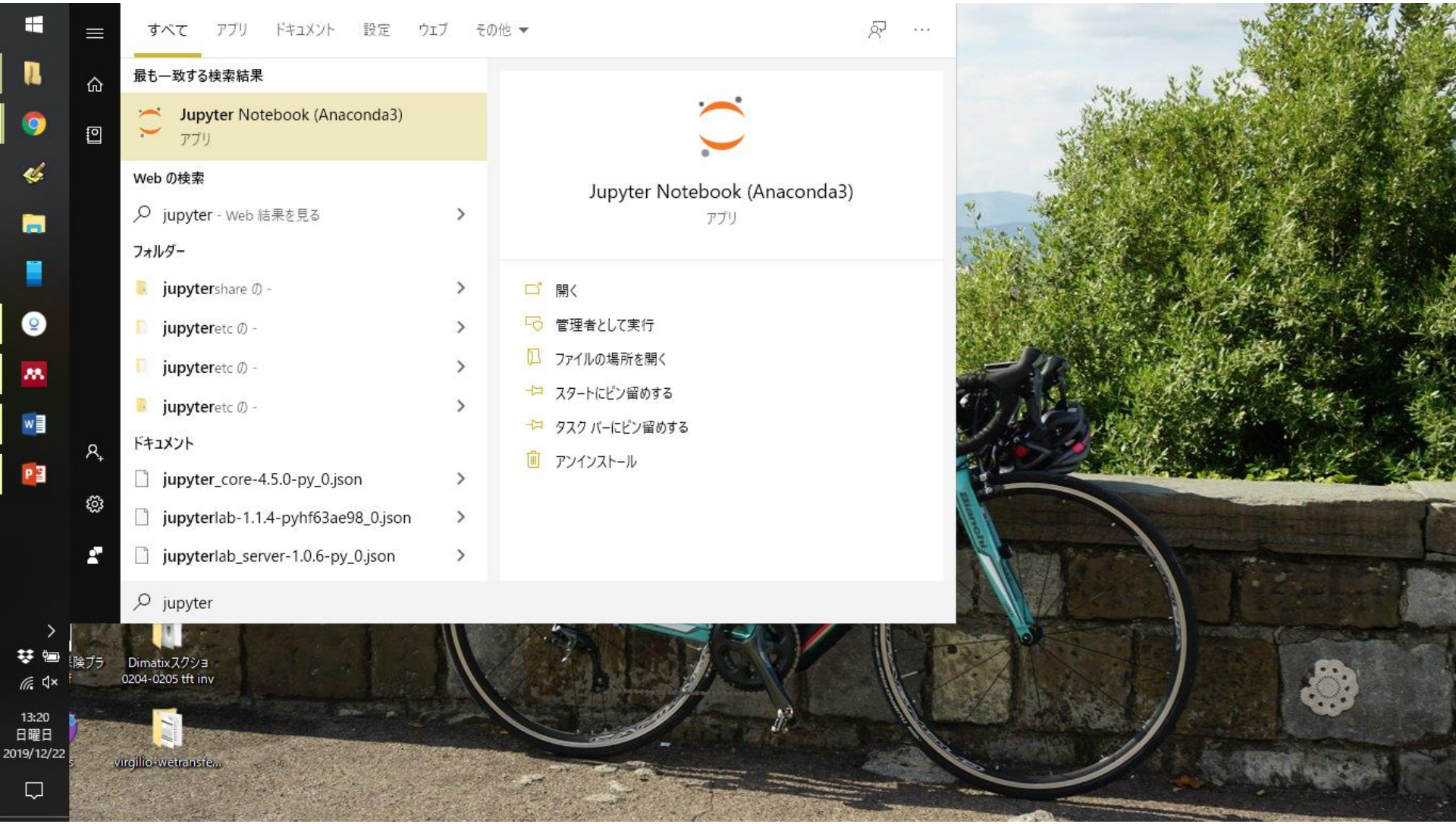
Plotting

Signal processing

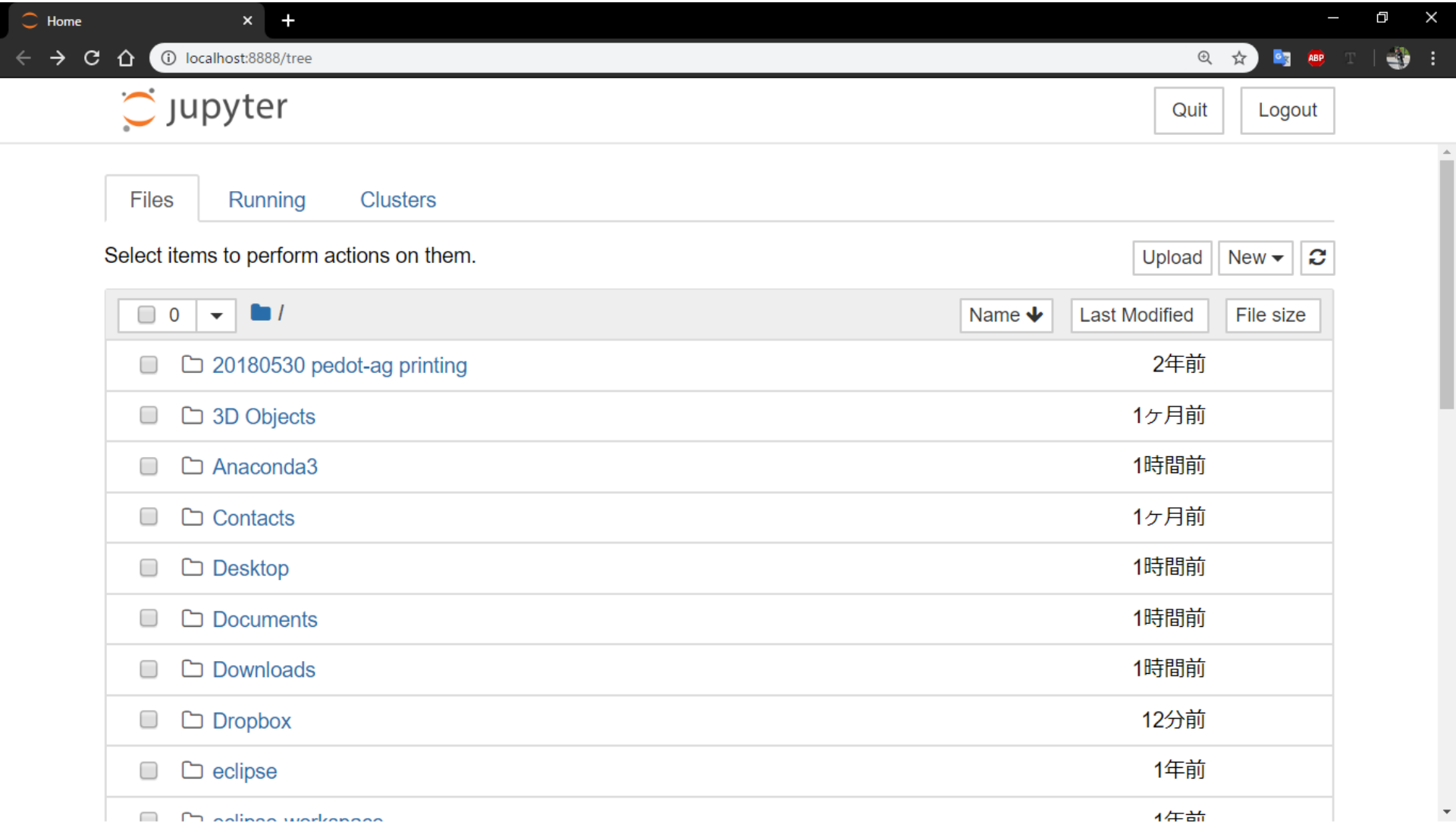
Statistics

Database etc.

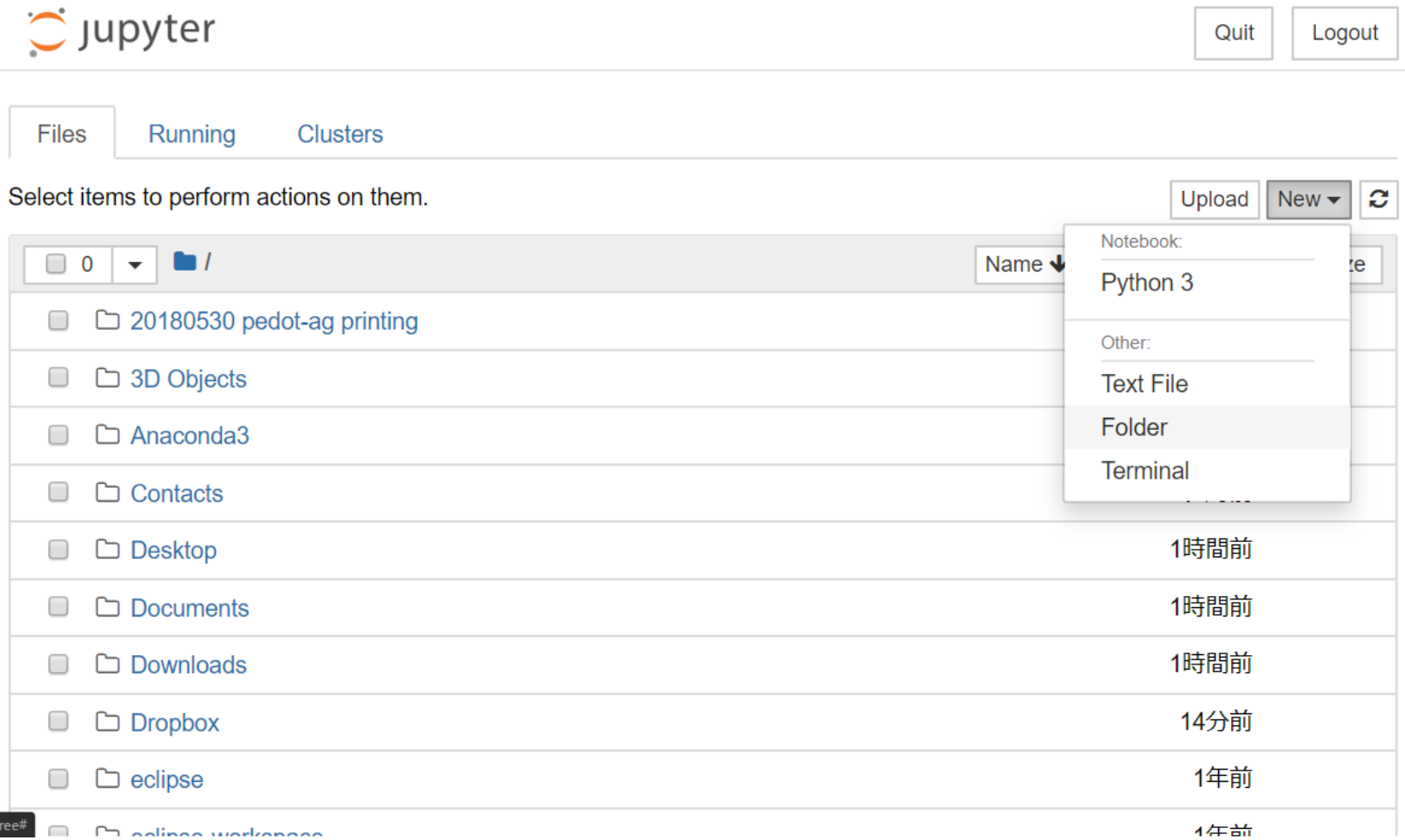
Jupyter: coding interface



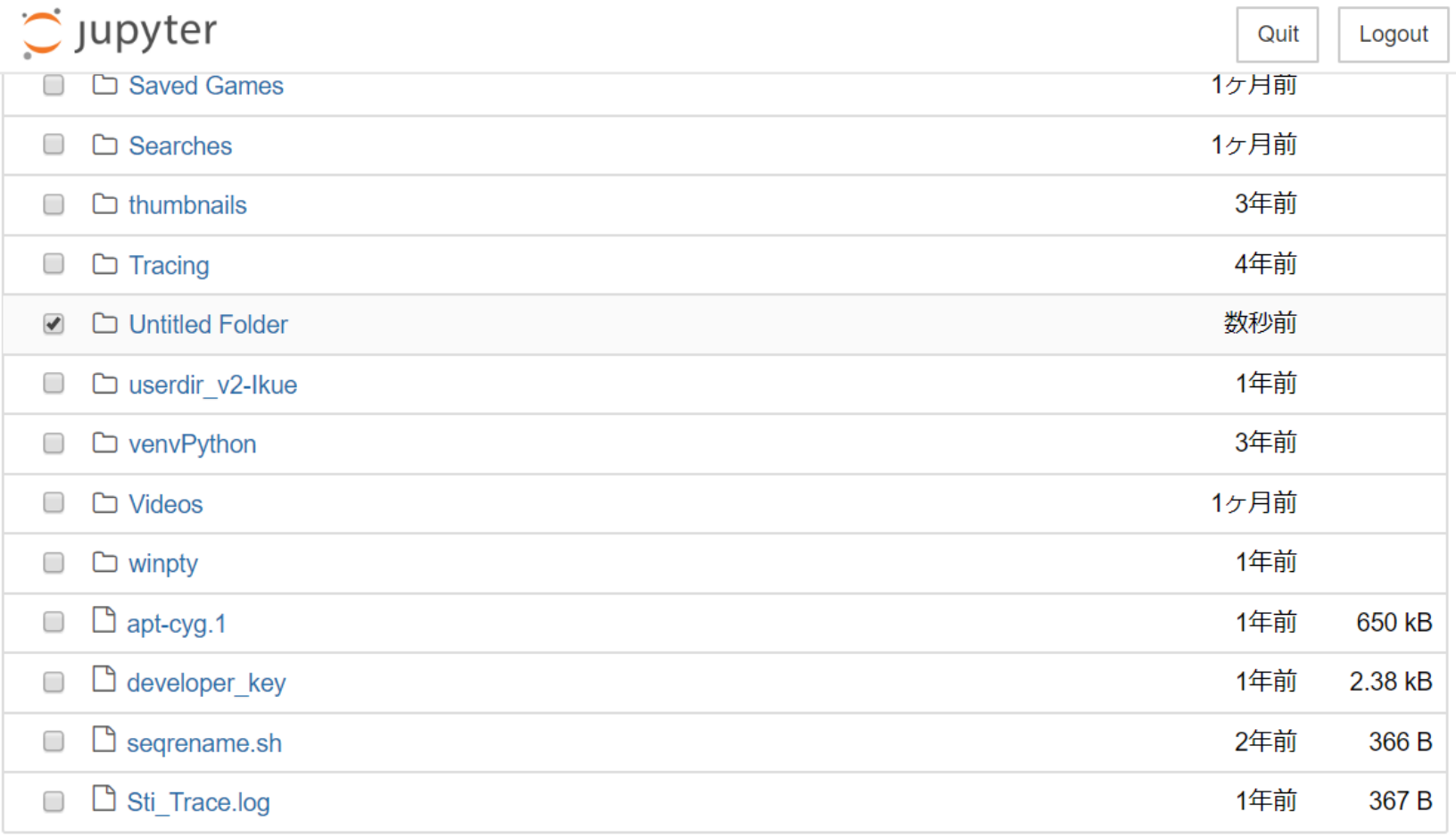
Jupyter: coding interface



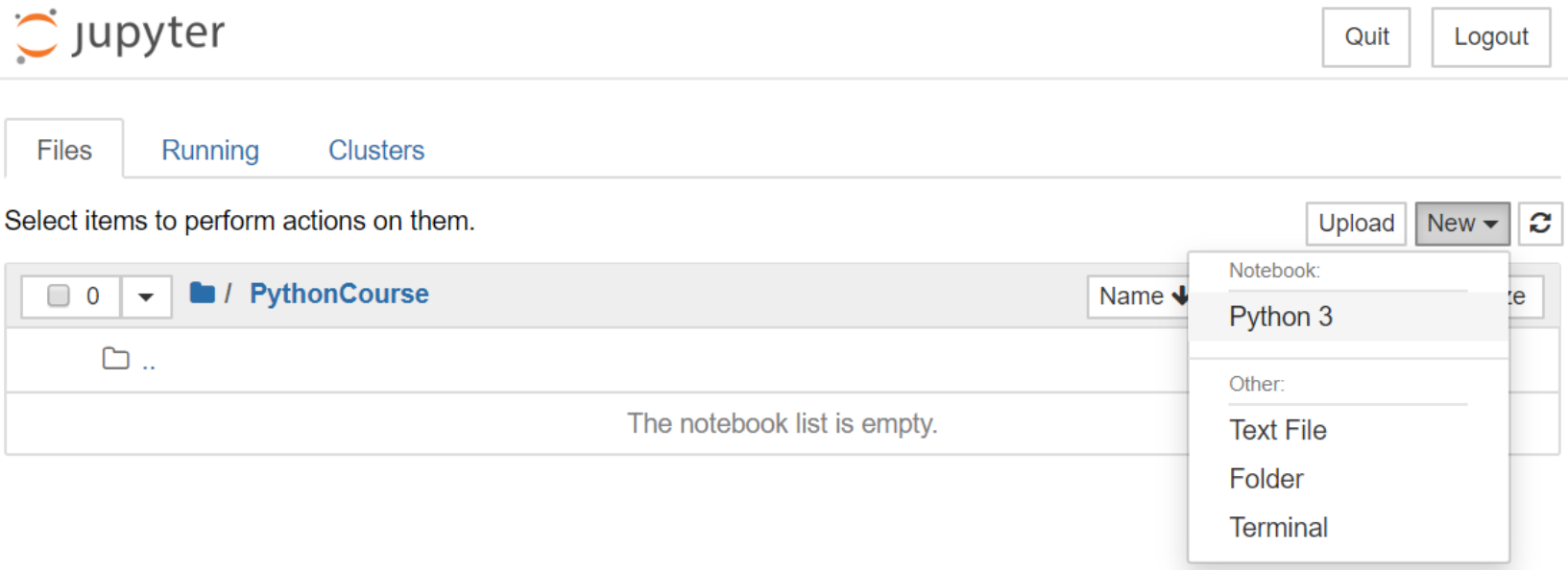
Create dedicated folder



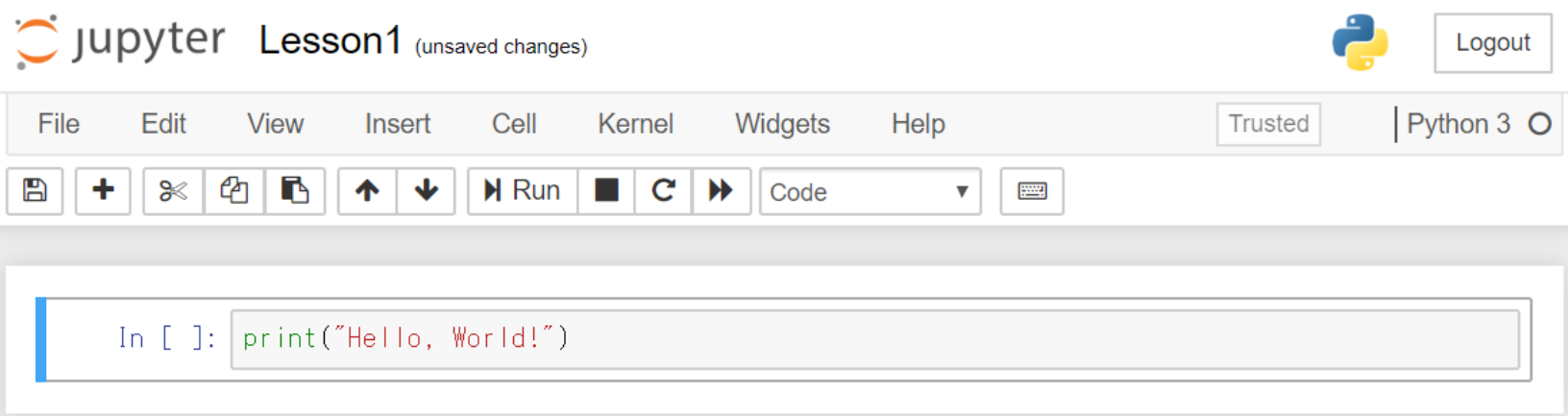
Rename


jupyter
Quit
Logout
Saved Games 1ヶ月前
Searches 1ヶ月前
thumbnails 3年前
Tracing 4年前
Untitled Folder 数秒前
userdir_v2-Ikue 1年前
venvPython 3年前
Videos 1ヶ月前
winpty 1年前
apt-cyg.1 1年前 650 kB
developer_key 1年前 2.38 kB
seqrename.sh 2年前 366 B
Sti_Trace.log 1年前 367 B

Create Python Notebook file

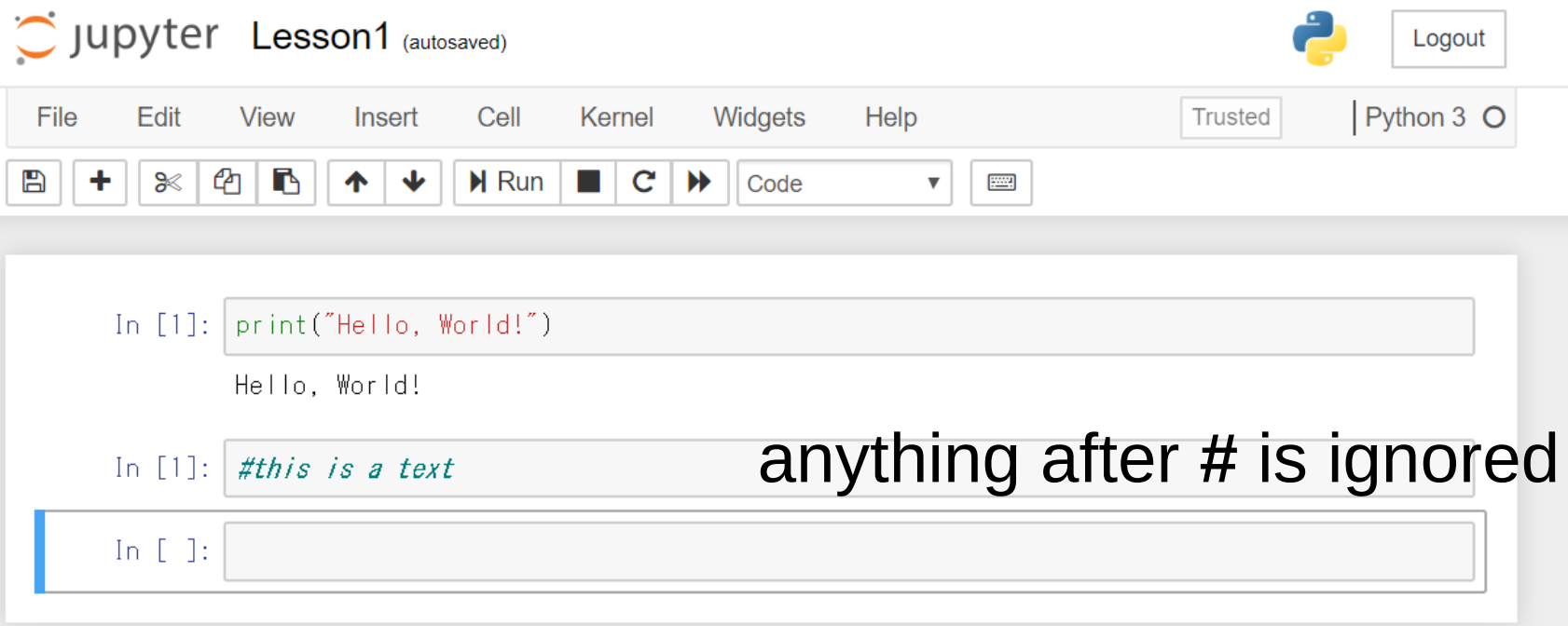


Let’s write your first code



Ctrl + Enter: execute the code in the cell
Shift + Enter: add a cell below
Double click: edit a fixed cell

Comment


jupyter Lesson1 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
In [1]: print("Hello, World!")
Hello, World!
In [1]: #this is a text
In [ ]:
anything after # is ignored

Syntax

```
print('Hello, World!')
```

Command

Options

Options can be changed

```
print(‘Hello, students!’)

print(‘Hello, colleagues!’)

print(‘Hello, Alice!’)

print(‘Hello, Bob!’)

print(‘Hello, Charlie!’)

print(‘Hello, Dave!’)
```

Suppose a greeting message

In [3]:
```
# greeting message
print("Hello, students!")
```

```
Hello, students!
```

In [6]:
```
print("Hello, colleagues!")
```

```
Hello, colleagues!
```

In [7]:
```
print("Hello, Alice!")
```

```
Hello, Alice!
```

Errors

```
In [7]:
print("Hello, Eve!)
```

```
  File "<ipython-input-7-855bf46aae91>", line 1
    print("Hello, Eve!)
                       ^
SyntaxError: EOL while scanning string literal
```

How to code *better*

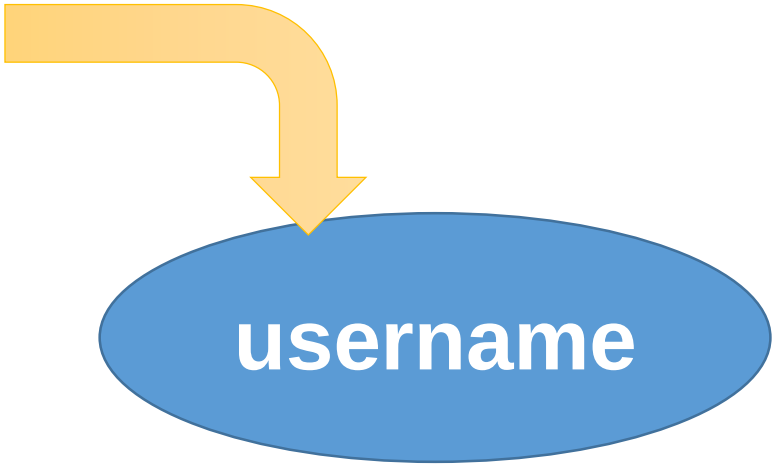
Eliminate redundancies

Where's redundancy?

```
print('Hello, students!')

print('Hello, colleagues!')

print('Hello, Alice!')

print('Hello, Bob!')

print('Hello, Charlie!')

print('Hello, Dave!')
```

Variables

students
colleagues
Alice
Bob
Charlie
Dave




```
print('Hello,
```


```
!')
```

Define a variable

```
username = ‘Alice’

print(f‘Hello, {username}!’)
```

f-String

Following keywords are not permitted:

```
and, as, assert, break, class, continue, def, del,
elif, else, except, exec, finally, for, from,
global, if, import, in, is, lambda, not, or, pass,
print, raise, return, try, while, with, yield
```

Let’s do it!

File Edit View Insert Cell Kernel Help

Trusted | Python 3

In [16]:
```
username = "Alice"
print(f"Hello, {username}!")
```

```
Hello, Alice!
```

In [17]:
```
username = "Bob"
print(f"Hello, {username}!")
```

```
Hello, Bob!
```

In [18]:
```
username = "Charlie"
print(f"Hello, {username}!")
```

```
Hello, Charlie!
```

Variable type

`int`	a = 1
`float`	a = 1.2398537
`bool`	a = True | a = False
`str`	a = “Alice”

int/float Arithmetic

File Edit View Insert Cell Kernel Help

Trusted | Python 3

In [24]:

```
a = 1
b = 3
c = 5
d = 11
print(a + b)
print(c - b)
print(b * c)
print(d / c)
print(d % b)
```

```
4
2
15
2.2
2
```

bool

Boolean, ‘True’ or ‘False’, without ‘ or “

In [26]:
```
x = True
y = False
print(x)
print(y)
```

```
True
False
```

str

File Edit View Insert Cell Kernel Help

Trusted | Python 3

In [3]:

```
st1 = "Hello"
st2 = ", World!"
print(st1[0])
print(st1[1])
print(st1[2])
print(st1[3])
print(st1[4])
print(st1 + st2)
```

```
H
e
l
l
o
Hello, World!
```

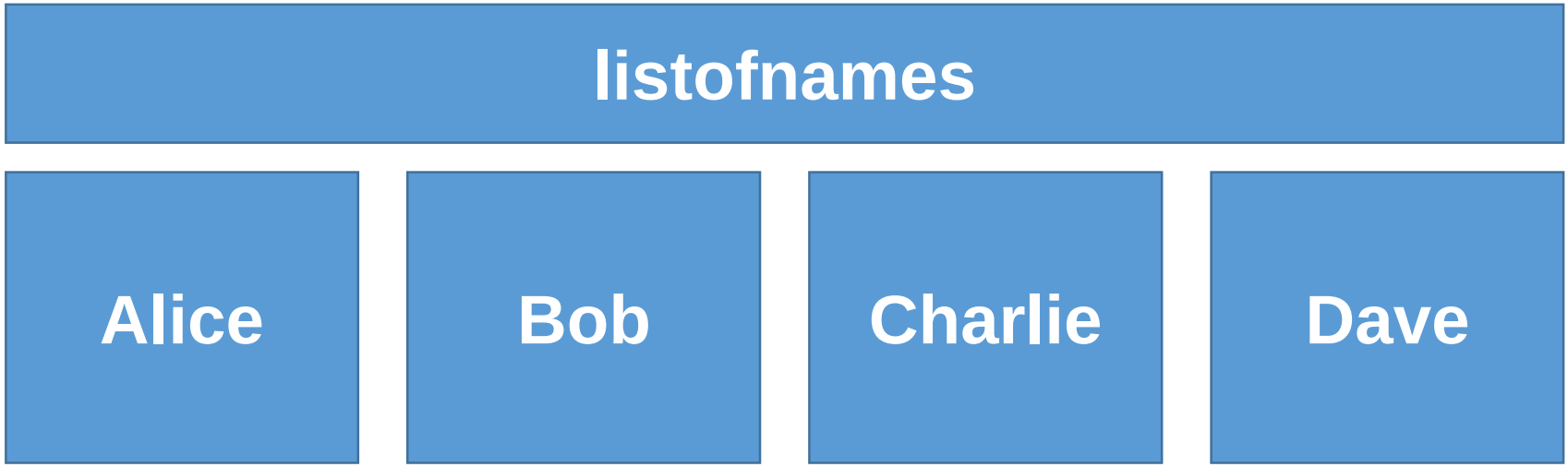
How to code *better*

Eliminate repetitions

Make a `List`

Never forget **COMMAS!**

```
listofnames = [“Alice”, “Bob”,
                “Charlie”, “Dave”]
```



Use a `for`/`in` loop

In [15]:

```
listofnames = ["Alice", "Bob", "Charlie", "Dave"]
for username in listofnames:
    print(f"Hello, {username}!")
```

```
Hello, Alice!
Hello, Bob!
Hello, Charlie!
Hello, Dave!
```

Compound types

List

```
listofnames = [“Alice”, “Bob”,,,]
```

Tuple

```
group = (“Alice”, “Bob”,,,)
```

Dictionary

```
ages = {“Alice”:14, “Bob”:18,,,}
```

List

Element is associated with `index`

`index` starts from 0

In [19]:
```
ic = ["Ancona", "Bolgona", "Como", "Domodossola", "Empoli"]
ic = ic + ["Firenze", "Genova", "Hotel", "Imora"]
print(ic[0])
print(ic[3])
print(ic[-1])
print(ic[-3])
```

```
Ancona
Domodossola
Imora
Genova
```

List

Part of list can be extracted by slice

`list[start:end:step]`

In [21]:
```
print(ic[1:2])
print(ic[3:])
print(ic[::2])
print(ic[::-1])
```

```
['Bolgona']
['Domodossola', 'Empoli', 'Firenze', 'Genova', 'Hote
l', 'Imora']
['Ancona', 'Como', 'Empoli', 'Genova', 'Imora']
['Imora', 'Hotel', 'Genova', 'Firenze', 'Empoli', 'Dom
odossola', 'Como', 'Bolgona', 'Ancona']
```

Dictionary

Element is associated with key

key can be non-integer

In [16]:

```
ages = {"Alice":14, "Bob":18, "Charlie":30, "Dave":90}
print(ages["Alice"])
print(ages["Dave"])
```

```
14
90
```

Comparison

==	equal to
!=	not equal to
>	larger than
<	smaller than
>=	equal or larger than
<=	equal or smaller than

Conditional statement: `if`

In [9]:

```
a = 1
b = 2
if a != b:
    print("Of couse 1 is not 2.")
else:
    print("Mathematical common sense has been overturned!")
```

```
Of couse 1 is not 2.
```

In [6]:

```
if "cat" == "dog":
    print("Cat is dog")
elif "cat" == "mouse":
    print("Cat is mouse")
else:
    print("Cat is not a dog nor mouse")
```

```
Cat is not a dog nor mouse
```

Control flow

`for a in b :` process each element in b

`while A :` repeat process while A is True

In [3]:

```
i = 0
while i < 5:
    print(i)
    i = i + 1
```

```
0
1
2
3
4
```

Break a control flow

In [31]:

```
h = 8
while h < 24:
    print(f"{h}.00 working")
    h = h + 3
    if h > 22:
        break
print("I'm too tired!")
```

```
8.00 working
11.00 working
14.00 working
17.00 working
20.00 working
I'm too tired!
```

User-made function

File Edit View Insert Cell Kernel Help

Trusted | Python 3

In [25]:
```
def func():
    print('this is a test function')
func()
```

```
this is a test function
```

In [27]:
```
def fahrenheit(T_in_celsius):
    # returns the temperature in degrees Fahrenheit
    return (T_in_celsius * 9 / 5) + 32
print(fahrenheit(25))
```

```
77.0
```

Today's Summary

Variables	`int`	0, 1, 2, 3…
	`float`	1.4396372
	`bool`	True/False
	`str`	"Alice"
Compounds	`list`	[0, 1, 2, 3, 4…]
	`tuple`	(0, 1, 2, 3, 4…)
Comparison	`if A:`	==, !=, >, <, >=, <=
Control flow	`for A in B:`	`while A:` `break`
Function	`def nameOfFunction():`	

Exercise 0

1. Print the natural numbers from 1 to 100.
2. Print the square of natural numbers from 1 to 100.
3. Print the square of unique numbers from 1 to 100.
4. Print the square of natural numbers that are not multiples of 3 from 1 to 100.
5. Print power of 2 within the range of 1 to 5000.
6. Print cube root of natural numbers from 1 to 10 to the fourth decimal place.

Advanced:

Do above in 1 line for each using list comprehension

Exercise 1

Branch: master ▾ CodingSeminarCMBR / Lesson1_Exercise1.ipynb Find file Copy path

ikuehirata exercise updated 9cdbbdb 17 seconds ago

1 contributor

50 lines (50 sloc) | 1.86 KB Raw Blame History

Exercise 1

1. Print an invitation for a drink to the people in the lists following the conditions below;
2. Each person answers yes or no. Print the name of the attendee and count how many people attend.

Conditions

1. Names are stored in the list `names`, ages in `ages`, answers in `ans`;
2. Do not invite underaged (less than 16.)

In [1]:
```
names = ["Alice", "Bob", "Charlie", "Dave", "Eve", "Faythe", "Grace", "Heidi", "Ivan", "Judy", "Mallory", "Niaj", "Olivia", "Peggy", "Rupert", "Sybil", "Trudy", "Victor", "Walter"]
ages = {'Alice': 58, 'Bob': 10, 'Charlie': 27, 'Dave': 12, 'Eve': 50, 'Faythe': 35, 'Grace': 39, 'Heidi': 23, 'Ivan': 53, 'Judy': 28, 'Mallory': 49, 'Niaj': 33, 'Olivia': 17, 'Peggy': 38, 'Rupert': 55, 'Sybil': 9, 'Trudy': 40, 'Victor': 56, 'Walter': 37}
ans = {'Alice': 'Yes', 'Bob': 'No', 'Charlie': 'No', 'Dave': 'Yes', 'Eve': 'No', 'Faythe': 'No', 'Grace': 'No', 'Heidi': 'Yes', 'Ivan': 'Yes', 'Judy': 'Yes', 'Mallory': 'No', 'Niaj': 'No', 'Olivia': 'No', 'Peggy': 'Yes', 'Rupert': 'No', 'Sybil': 'Yes', 'Trudy': 'Yes', 'Victor': 'No', 'Walter': 'No'}
# invitation message example: "Hello Alice, let's grab some beer!"
```

Exercise 2

ikuehirata / CodingSeminarCMBR

Unwatch 1 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Actions | Projects 0 | Wiki | Security | Insights | Settings

Branch: master ▾ CodingSeminarCMBR / Lesson1_Exercise2.ipynb

Find file | Copy path

ikuehirata add l1-e2 — 76a71b1 1 minute ago

1 contributor

60 lines (60 sloc) | 1.27 KB

Raw | Blame | History

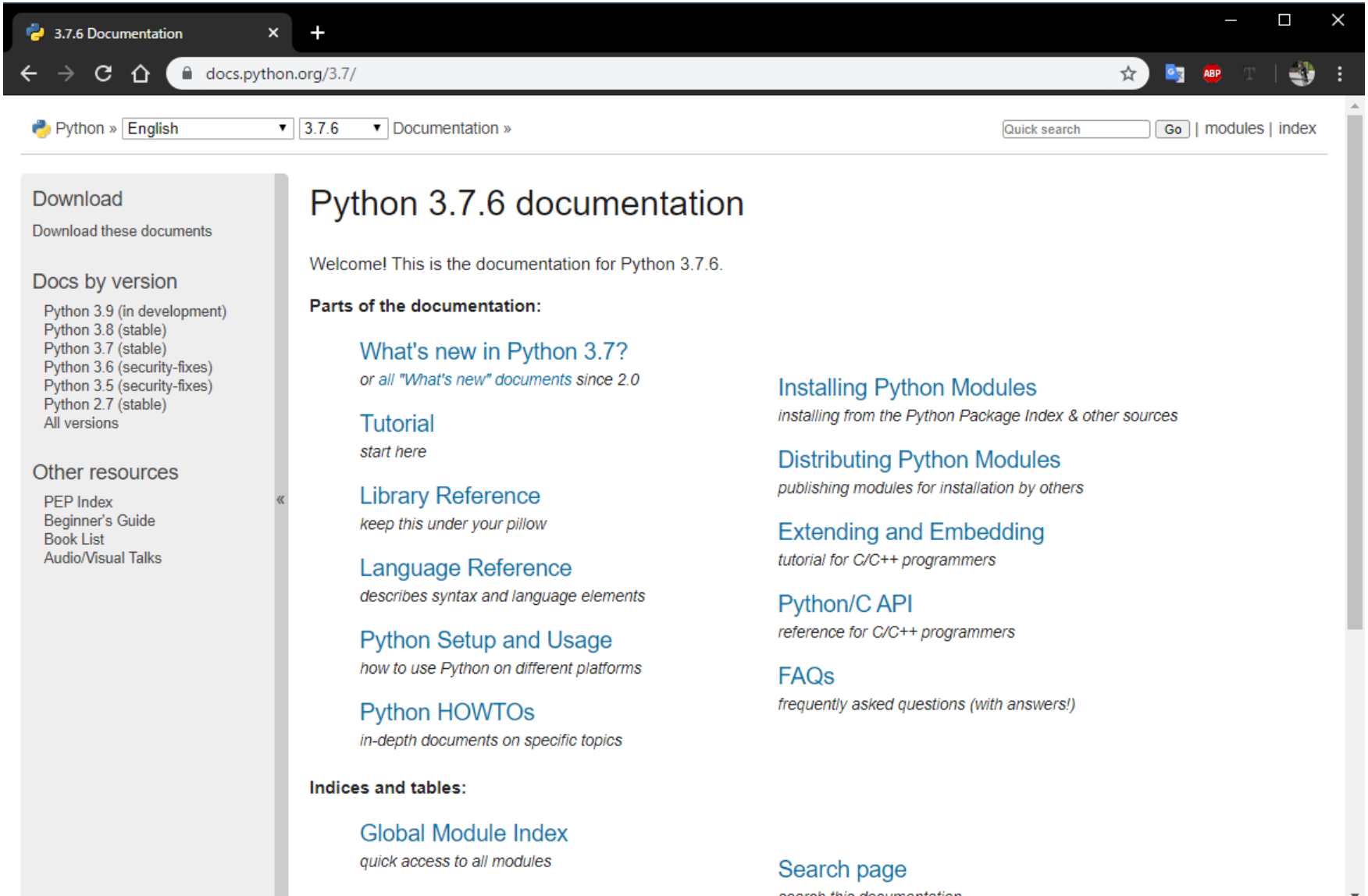
Lesson 1 - Exercise 2

Write a code that determins if the input number is a primal number or not, such as:

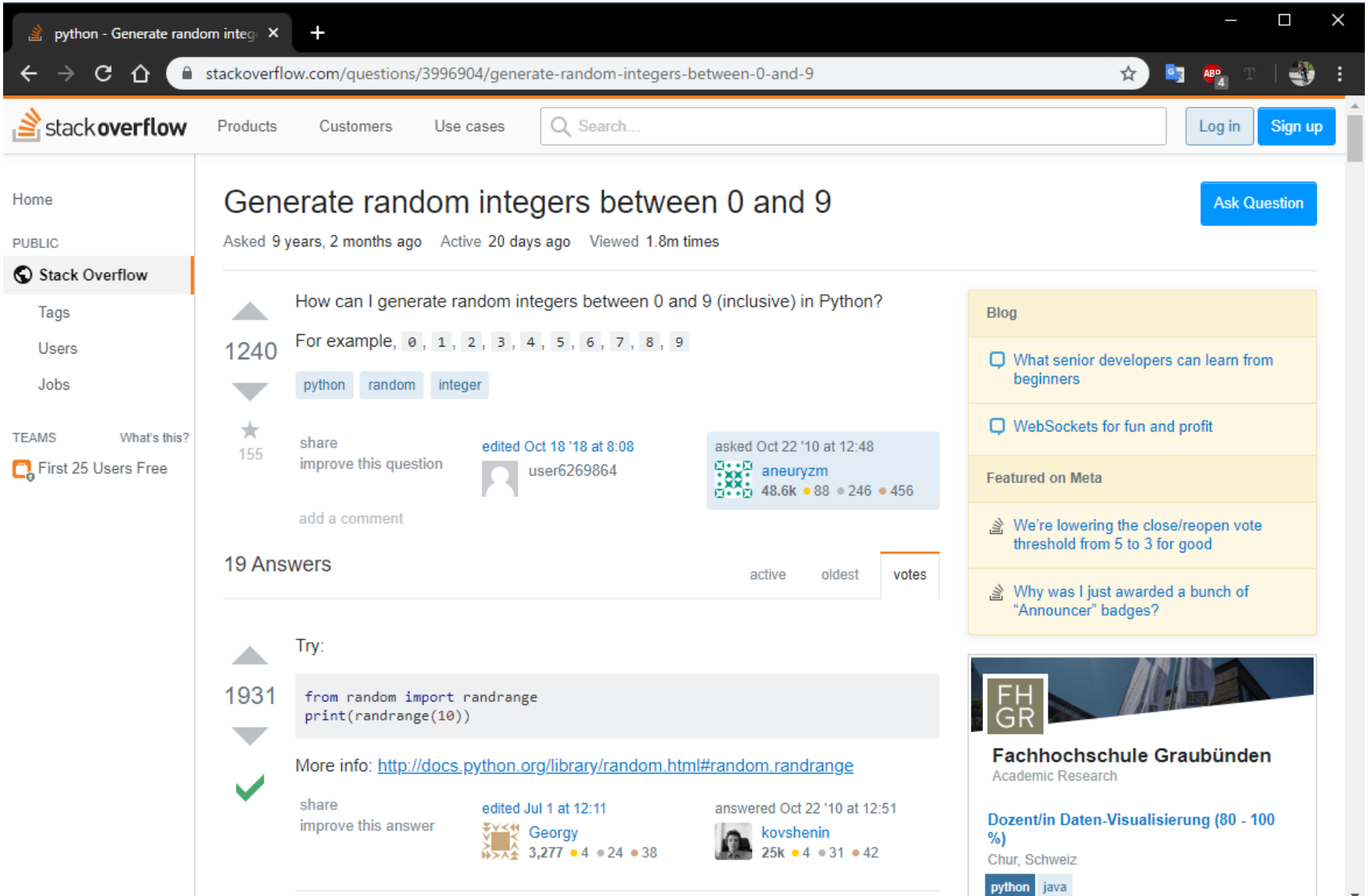
```
def isprime(n):
    your code here
    if (n is a prime number):
        return True
    else:
        return False

isprime(10)
isprime(297)
isprime(104593)
```

The output is False, ?, ? (find the answers by yourself!)

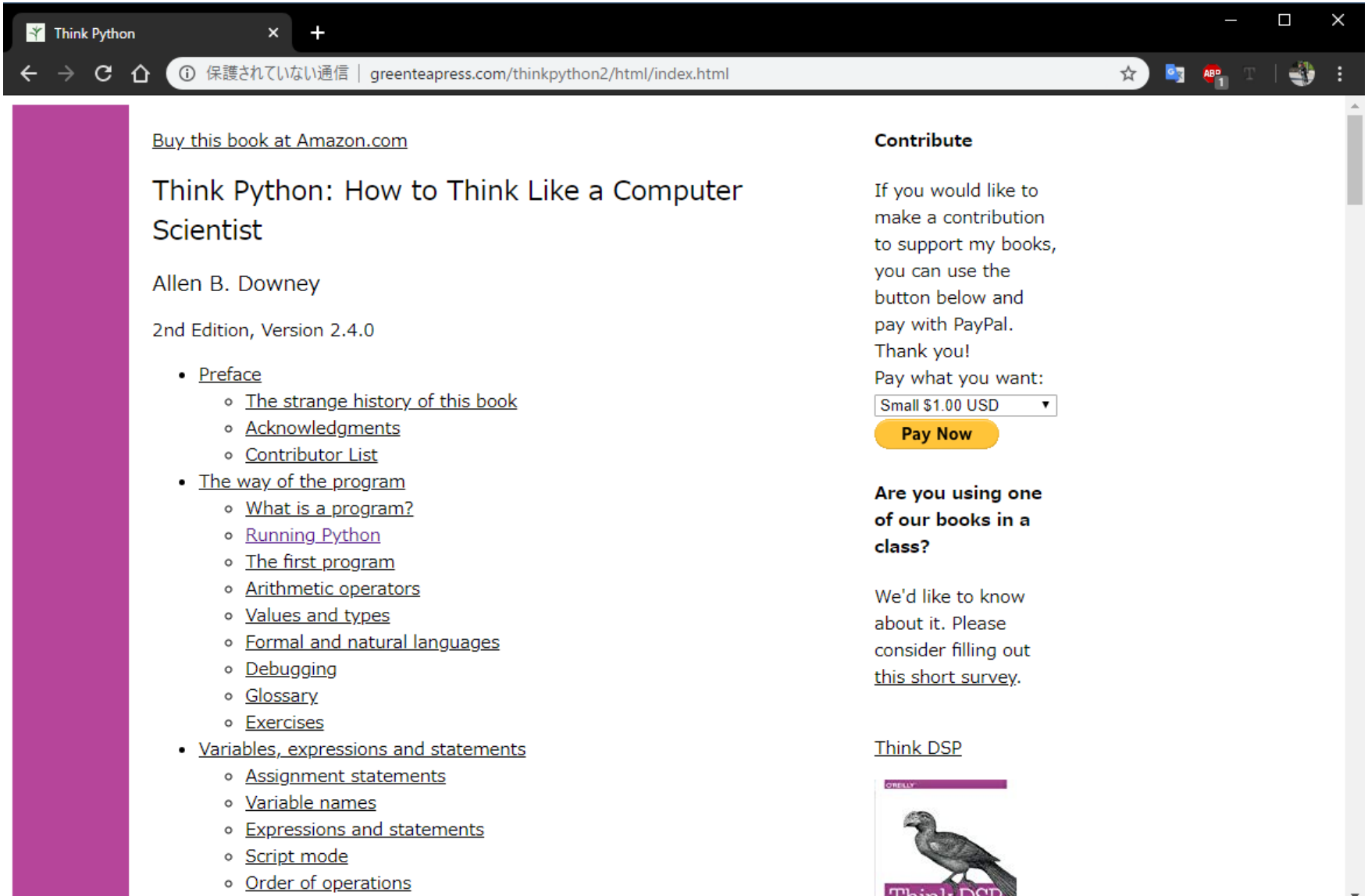
Your friend: official documentation



Your friend: Stack Overflow

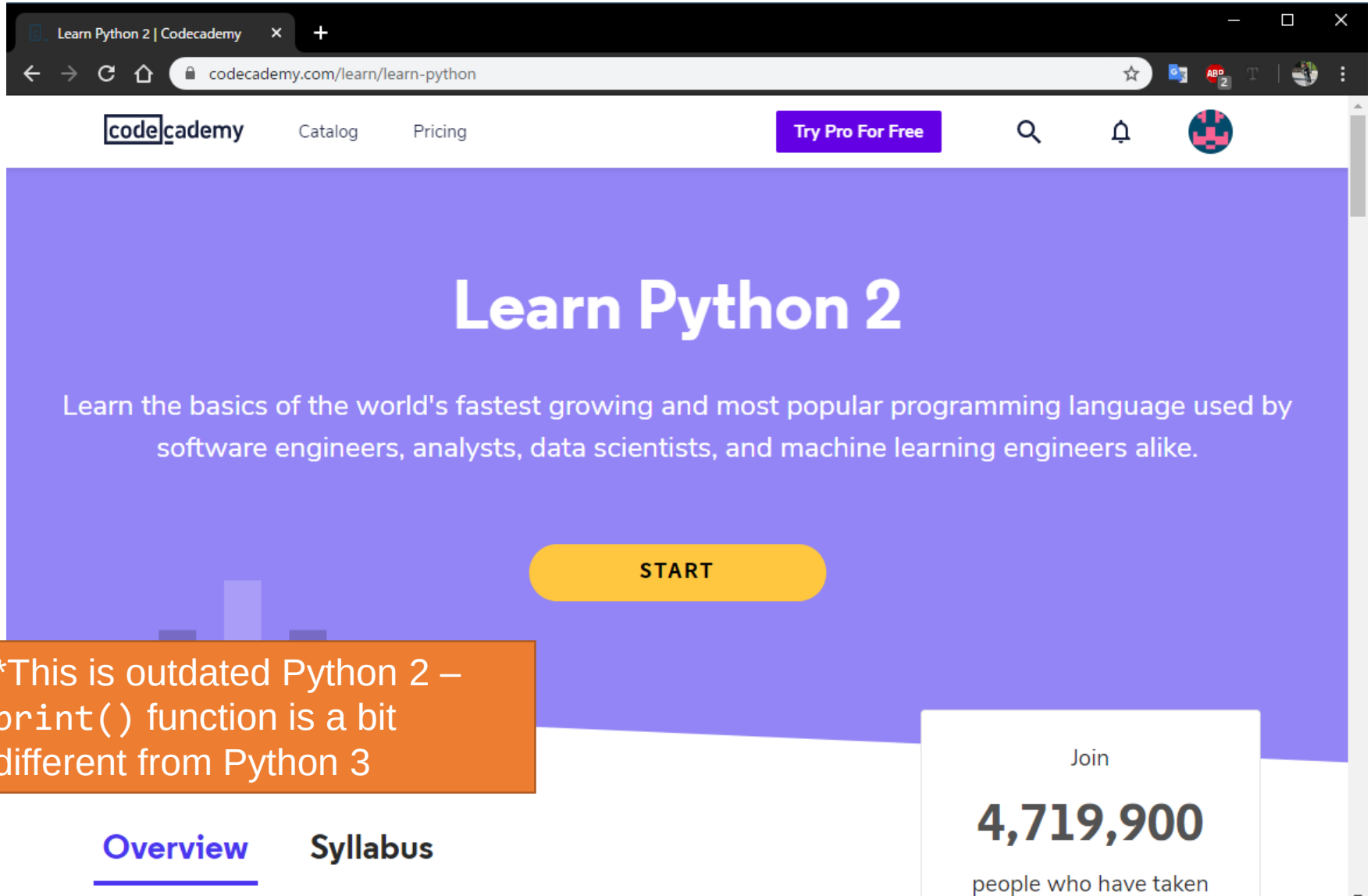


Your friend: other materials



Your best friend: Google

To study by yourself: codecademy



*This is outdated Python 2 – `print()` function is a bit different from Python 3