Introduction to Python Day 2

Verjinia Metodieva and Daniel Parthier

2025-02-04

Jupyter Notebook

Recap homework

Let's take a look at the homework

Functions part 2

Goal of today

```
import numpy as np
import os

def AP_check(folder):
    AP_sweep_count = 0
    for filename in os.listdir(folder):
        if filename.endswith('.csv'):
            with open(os.path.join(folder, filename), 'r') as file:
                data = np.loadtxt(file, skiprows=1)
                if any(data > 20):
                    AP_sweep_count += 1
                    print("AP found in " + filename)
                else:
                    print('No AP in ' + filename)
    return AP_sweep_count

file_path = 'data/sweeps_csv/'
```

```
sweep_count = AP_check(file_path)
print(sweep_count)
```

This is an example to showcase what we will achieve today.

Global vs. Local

Short interlude

- Whole numbers: Integers `int`

```
type(1)
```

```
int
```

- Real numbers: Floats `float`

```
type(1.0)
```

```
float
```

- Most of the time it might not matter[1]

```
1 == 1.0
```

```
True
```

- Sometimes there is a difference and we will see later why

Most of the time python handles the integer vs. float automatically. You will not have to worry about assigning.

[1] In python

Conditional statements

The important question of what to do "if" something happens.

- Programming languages are languages
- `if` something is `True`
  - you should do `something`
- `else`
  - do `something else`

```
if statement:
    print("the statement is true")
else:
    print("the statement is false")
```

This structure is the simplest of conditionals. The statement has to be `True` to enter the `if` part to execute. Should the statement by `False` it will skip and enter the `else` part which will then be executed.

Multiple if-statements

```
value = 3
if value == 1:                                                    ①
    print("the value is 2")
elif value == 2:                                                  ②
    print("the value is 2")
elif value == 3:                                                  ③
    print("the value is 3")                                       ④
else:
    print("the value is something else")
```

① Check if `value` is `1`
② Check if `value` is `2`
③ Check if `value` is `3`
④ Execute block

```
the value is 3
```

Statements will be checked sequentially. Should one statement be `True` the corresponding part of the `if/elif` block will be executed. All other blocks after that will be skipped. This means one `True` expression is enough.

Short forms for conditionals

```
amplitude = 24
is_action_potential = "is AP" if amplitude > 0 else "no AP"
print(is_action_potential)
```

```
is AP
```

- You can write a lot on one line
  - Do if you have to but be careful

How to check if everything is true?

- Validate all of the statements in a list

```
everything_is_true = [True, True, True]
something_is_true = [True, False, False]

all(everything_is_true)
all(something_is_true)
```

```
True
```

```
False
```

- Sometimes only something has to be true

```
any(everything_is_true)
any(something_is_true)
```

```
True
```

```
True
```

For loops

Enumerate

Range

List comprehension

Compare different functions

While loops

- Perform a task `while` something is `True`
- Be careful:
  - Some loops never finish (get stuck)
  - Make sure that condition for ending the loop can be fullfilled

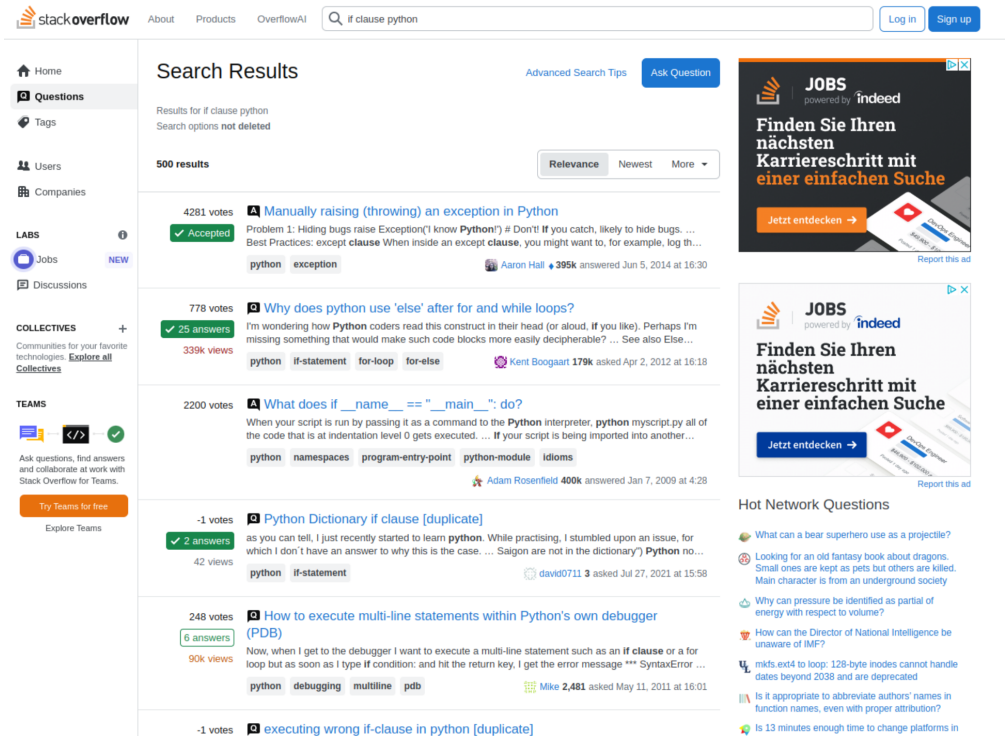
```
while check_condition:
    perform_task()
```

Errors and how to read them

There are useful resources regarding errors

- Simply googling works surprisingly well
- You will often end up on stackoverflow
  - There is no question which was not already asked[2]

[2] if that is not true open up a question



Types of errors

1. SyntaxErrors
2. NameError
3. TypeError
4. IndexError
5. AttributeError
6. etc.

Fix errors