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

```
# TODO: add useful example to introduce class
# Some function we have to decide
def useful_function(par1, par2, par3):
    # some loop
    for i in par1:
        if par1 == "something":
            # do something
        else:
            pass # or something else
    return None # or an object
```

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.

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`

[1] In python

```
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.

How to check if everything is 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

Types of errors

Fix errors