Exercise class 1

(week 8)

Introduction to Programming and Numerical Analysis

Class 8 and 4

Rosa Haslund Meyer

Spring 2024

KØBENHAVNS UNIVERSITET

Functions

Exercises

Classes

Exercises

Floating point numbers and **NumPy basics** will make sense when you work through the DataCamp exercises.

However, if you have any questions or uncertainties along the way, feel free to ask!

Plan for today's exercise class

- 15.15-15.30: recap of important concepts from the week's lecture – functions
- 15.30-16.15: (including a break): exercises – individual/group work on DataCamp/installations/problem set 0 if you haven't looked at it already
- 16.15-16.30: recap of important concepts from the week's lecture – classes
- 16.30-17.00: exercises – individual/group work

Functions in Python

Whenever we need to perform the same operation multiple times, we typically use **functions**:

- A function is a block of code which only runs when called
- A function performs a specific task and can easily be reused/repeated
- A function takes **inputs**, performs an operation, and returns an **output**

Reasons for using functions:

- It's generally bad practice to repeat code more than once
- If you ever have or need some operation multiple times, you should define a function
- Makes the code more manageable and thus easier for outsiders/group members to read and understand your code

"Python Data Science Toolbox (Part I)" is all about functions!

Global/local scope

Variables in Python can exist in either **global** or **local** scope:

- A variable is only available from inside the "region" it is created

Global scope:

- When a variable is created inside the **main body** of the Python code it's a global variable and belongs to the global scope
- Global variables are **available from any scope**: both global and local scope
- As such, global variables are kept in memory **and can always be accessed** when your code is running

Local scope:

- A variable created inside any given function belongs to the local scope of that given function, and **can only be accessed within that function**
- As such, local variables are **deleted from memory** when the function has terminated running

Last time for DataCamp!

You must have completed the following 4 courses from DataCamp by **February 25th**:

- Introduction to Data Science in Python
- Intermediate Python
- Python Data Science Toolbox (Part I)
- Python Data Science Toolbox (Part II)

The first two exercises classes are dedicated to working on DataCamp – but expect to do a lot of work on it at home as I want to make sure you understand the basic concepts and get all the installation done.

Don't rush through the exercises – focus on understanding the code to make a good foundation. Understanding the basics is paramount for solving the later problem sets and assignments.

If you have trouble understanding any concepts, please let me know and I'll be happy to help.

If you get an error message or forgot some syntax, try to Google it before asking. When it comes to programming, Google (and StackOverflow) can solve a lot of minor and syntax errors!

Time for exercises

Your tasks:

- (If you haven't already done so, follow the installation guides)
- (Problem set 0)
- Work on DataCamp

If this is your very first-time programming or working Python and your feeling overwhelmed, I can recommend to look at some of the exercises in DataCamp introducing Python such as "Introduction to Python", which gives a gentle introduction to data types and variables.

At **16:35**, I'll do a recap on **classes** and give a short demonstration of how to use classes to write manageable and neat code.

Classes

We have seen many different types of variables, each with different attributes and methods:

- `int, float, list, dicts, np.array`

Think of **classes** as user-defined types:

- A class can be viewed as an advanced "container" that contains both data (attributes) and functions specific to the class (methods)
- The specific data in a class originates from creating an instance of the class – an object/variable
- Objects get their variables and functions from classes and can be view as an encapsulation of variables and functions into one "entity" – classes is a kind of **"template"** used to create objects

As such, we can now create our own types of variables using **classes**:

- We can choose and define which attributes and methods our class has
- Using classes when coding keeps everything manageable and nicely structured

Next time...

Video lectures:

- Print
- Plot
- Optimize

Exercises – Problem set 1. Solving the consumer problem:

- Functions (lambda functions)
- Loops and `if-statements`
- NumPy tools
- Plotting and printing
- Optimization with and without constraint
- Numerical optimizers/solvers
- (Classes)

Questions and/or comments?