Exercise class 1

Introduction to Programming and Numerical Analysis

Class 3 and 6

Annasofie Marckstrøm Olesen

Spring 2024

UNIVERSITY OF COPENHAGEN

Welcome!

Tips for DataCamp and Installation

Exercises

Atomic types

Containers

Conditionals and loops

More exercises

Welcome!

About me

- Annasofie Marckstrøm Olesen
- PhD student at CEBI
- Took this course 2022 and TA'ed in 2023
- Always open to feedback about what works and what doesn't
- Can be contacted at **aso@econ.ku.dk** or through Absalon.

The exercise classes

Lectures are mostly in video format, so exercise classes are your main chance for asking questions in person.
You need to be active learners - important that you get your hands dirty and learn to locate and fix mistakes!

Plan for exercise classes

- I give a recap of important concepts from this week's lecture and tips for the problem set.
- Individual/group work on problem set.
- 5 min recap

Course site, Absalon and Github?

Here are a few pages you need to know:

- Course website
- Letures Github repository
- Exercises Github Repository

I will put material relevant to the exercise classes in our own Github repository here. Please contact me if you haven't received a link!

We will not be using Absalon a lot in this course.

Tip for installation

You'll need to install:

- **Anaconda**: A Python distribution, which helps you download Python and packages.
- **Git**: A version control system that keeps track of changes to files and allows you to cooperate on projects online.
- **Visual Studio Code**: A text editor/IDE where you can write and run code files and notebooks.

Tips for installation

Follow the instructions on the course website carefully and in the right order.

Make sure you are installing Anaconda on a path with no spaces or special characters.

If something fails, the best approach is often to restart the program - or even uninstall and start from scratch.

Installation may take some time - you can do DataCamp while waiting.

DataCamp

By **February 25th** you must have completed 4 courses in DataCamp:

- Introduction to Data Science in Python
- Intermediate Python
- Python Data Science Toolbox (Part 1)
- Python Data Science Toolbox (Part 2)

The first two exercise classes are dedicated to working on DataCamp, but you should expect to do some of the work at home.

Try not to speed through the exercises - focus on understanding the code and why it works. Understanding the basics is necessary for solving the problem sets and the assignments.

If you have trouble understanding a concept, let me know and I'll be happy to help.
If you get an error code or if you forget the syntax, try to Google before asking me.

Time for exercises

Your tasks:

- Install according to guide
- DataCamp
- Problem set 0

If this is your very first time working with Python and you're feeling overwhelmed, I can also recommend the DataCamp course "Introduction to Python" for a gentle introduction to variables and data types.

At **16.15**, I'll do a recap on **atomic types, containers and loops**.

What is a variable?

A variable **points** to a piece of **data** stored on **memory**.


Uni
DEFINICLAS
año

Atomic types

Four **atomic types**:

- Booleans (`True`, `False`)
- Integers (1, 0, -3, 1582, ...)
- Floating point (0.1, 3.141592.., 2.71828..., ...)
- Strings (`'abc'`, `'123'`, `'hello world'`, ...)

Atomic types don't keep their reference when changed. They are "put in a new folder" instead.

Containers

Containers can be comprised of several atomic type variables:

- Lists, `[]`
- Dictionaries, `{}`
- Tuples, `()`
- Pandas DataFrames, `pd.DataFrame()`
- NumPy arrays, `np.array()`
- ...

Containers can be changed and keep their reference! Can lead to bugs, if you're not aware of it...

(Tuples are an exception as immutable.)

In the DataCamp courses

Introduction to Data Science in Python touches on different atomic types.

Intermediate Python introduces dictionaries and Pandas DataFrames.

Conditions

Conditions consist of `if`, `elif` or `else` followed by an expression which evaluates to a **boolean** type:

```
if x>0:
    print('x is positive')
elif x<0:
    print('x is negative')
else:
    print('x is 0')
```

Boolean expressions may also include boolean operators: `and`, `or`, `not`, `is`

Loops

Computers are really good at performing **the same task over and over**. We can use loops!

- `for`-loops for when you want a **specific number of iterations**. Eg. iterate over elements in a list, the number of variables in a data set, the number of time periods in a dynamic model etc.
- `while`-loops for repeating a process **until some condition holds**. Eg. evaluate function until it converges.

DataCamp course *Intermediate Python* explains conditionals and loops. List comprehensions are covered in *Python Data Science Toolbox part 2*.

Next time...

Video lectures:

- Functions
- Floating point numbers
- Classes
- NumPy Basics

Exercises

- DataCamp + Problem set 0