# Week 7

Introduction to Programming
and Numerical Analysis

Ebener, Luca

## Overview

- About me
- Exercise classes
- Programs
- DataCamp
- Takeaways week 7

## About me

**Headers**

- Last semester of master programme "Research in Economics and Finance" at University of Vienna
- Spent third semester on exchange at KU
- Interested in monetary and computational economics

**Contact**

- wmn553@econ.ku.dk

## Exercise Classes

- I present some key points of the week or hints how to tackle the problems
- You work on the problem sets alone or in groups and I go around answering questions
- Happy to answer your questions via mail
- You can send me topics you like to repeat at the beginning of the class in advance
- Open for suggestions

## Rest of the Semester

- First two classes: DataCamp and problem set 0
- Seven classes with problem sets
- Five classes dedicated to your projects
- Exam consists of three assignments during the semester and a final exam assignment

## Programs

**Installation**

- Follow the guide on the course webpage

**Programs**

- Anaconda: Comprehensive Python distribution with pre-installed packages and environments management for streamlined development.
- VS Code: Versatile and lightweight code editor with extensive Python support, including debugging capabilities and extensions for Jupyter notebooks integration.
- Git: Distributed version control system enabling collaborative development, tracking changes, and managing code repositories efficiently, crucial for team projects and versioning notebooks.

## DataCamp

**Exercises**

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

**General**

- Deadline is the 25 of February
- Do it with time as the estimated length of the courses is 15 hours
- Try to understand not only how to complete the problems, but why all the code works

## Atomic types

```python
x = 1 # x references an integer type object with a value of 1

type(x) # type of x
id(x) # address of x
sys.getsizeof(x) # amount of memory used by x

str(x) # parses into string
float(x) # parses into float
int(x) # parses into integer
bool(x) # pases into boolean

# arithmetic operators are e.g. +,y,/,**,%
# augmentation operators are e.g. +=,-=,*=,/=
# logical operators are e.g. ==,!=,<,<=,>,>=
```

## Containers

```
x = [0,1,2,3] # x references a list
x[1:2] # gives out the second to third element of the list
x[:-2] # gives out the first to second to last element of the list
# if you change the data of a list that one variable refers to, you
    change all variables referring to that list
y is x # true if they contain the same reference

y = copy(x) # y refers to a copy of list x not the same reference
y = deepcopy(x) # necessary, when the list contains mutable objects

z = (1,2,3) # z is a tuple which is an immutable list that cannot
    be changed

q = {'one': 1, 'two': 2, 'three': 3} # q is a dictionary
```

## Conditions and loops (1)

```python
if bool1:
    print('bool1 is true')
elif bool2:
    print('bool2 is true')
else:
    print('neither bool1 nor bool2 is true')

for x in xlist: # iterates over elements in xlist
    pass

[x*2 for x in xlist] # returns list where each element of xlist is
    doubled

while bool3: # executes until bool3 is false
    print('watch out for endless loops')
```

## Conditions and loops (2)

```python
for x in range(6): # iterates over list from 0 to 5
    pass

for i,x in enumerate(xlist): # x is the i'th element in xlist
    pass

for x,y in zip(xlist,ylist): # x is the element of xlist and y the
    element of ylist for every iteration
    pass

for key in my_dict.keys(): # iterates over the keys
    for val in my_dict.values(): # values
        for key,val in my_dict.items(): # key-value pairs
            pass
```