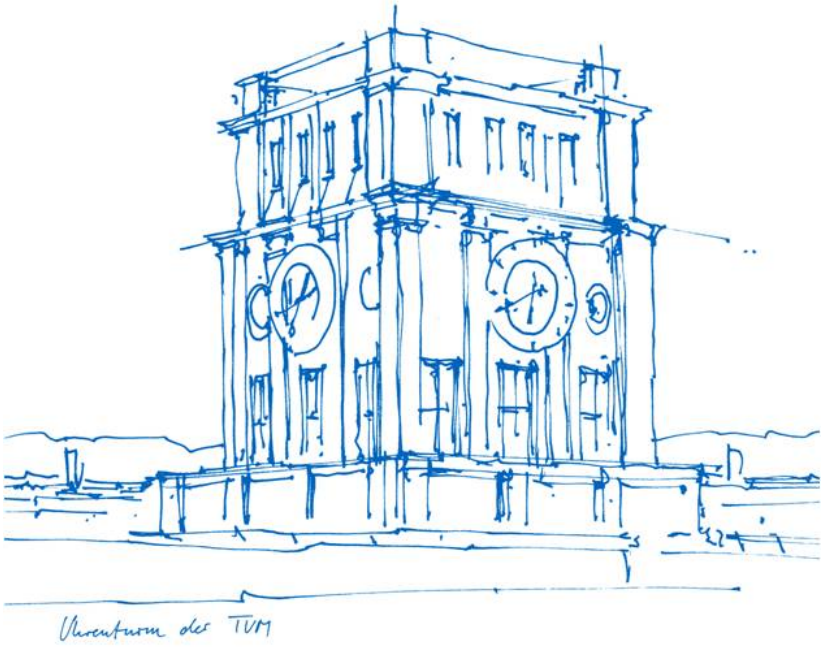
Exercises for
Social Gaming and
Social Computing
(IN2241 + IN0040) – Introduction to

Exercise Sheet 1
Introduction to Python



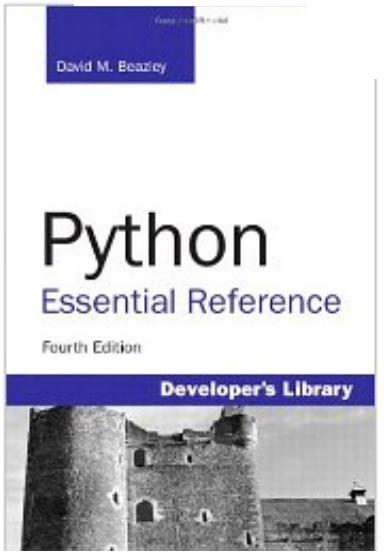
Exercise Content

Sheet Number	Exercise	Working Time
1	• Introduction to Python: basic Python programming language exercises • Graph Drawing using networkx, introduction to Steam API	Monday, May 27 - Monday, June 3, 24:00
2	• Centrality measures	Monday, June 3 - Monday, June 17, 24:00
3	• Collaborative Filtering Recommender System using Steam data	Monday, June 17 - Monday, June 24, 24:00
4	• Trying to infer toxic behavior from game data in DotA 2	Monday, June 24- Monday, July 01, 24:00
5	• Clustering EVE and BF2 players according to Radoff's motivational player types using K-means	Monday, July 01 - Monday, July 08, 24:00
6	• Analyzing short-term social context using mobile interaction data (Reality Mining)	Monday, July 08 - Monday, July 15, 24:00

Repetition: Python and IPython Books

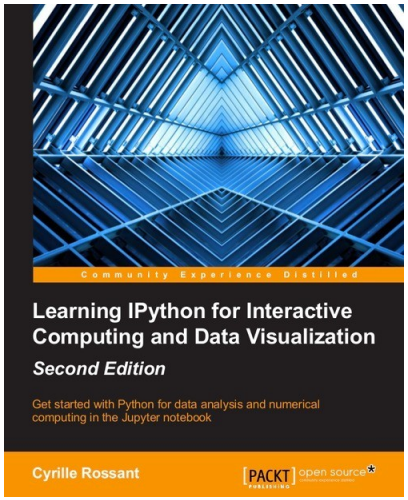
- **Learning Python:**
  Python Essential Reference (2012)
  by David M. Beazley, Safari Books
  (especially **chapter 1: A Tutorial Introduction (25 pages)**)

  free eAccess: https://eaccess.ub.tum.de/login



- **Learning IPython / Reference for IPython:**
  Learning IPython for Interactive Computing and Data Visualization (SECOND EDITION) by Cyrille Rossant, 175 pages, Packt Publishing, October 25 2015
  (Especially (free) chapter 1.4. A crash course on Python)

  free access:
  http://nbviewer.ipython.org/github/ipython-books/minibook-2nd-code/blob/master/chapter1/14-python.ipynb
  (do not try to open this ipynb with Jupyter directly. Instead, download all the ipynb's from the book from Github: https://github.com/ipython-books/minibook-2nd-code)



Repetition: Installation using Docker

- install Docker as a platform specific software
  - from Docker website
  - by Terminal/Command Line
- download the dockerfile provided to you on Moodle
- place it into the same folder as the exercises
- navigate to that folder and execute the following command:

```
docker build . -t arbitraryimagename
```

- this will install an Ubuntu as well as Python 3 and some libraries (numpy, scikit, pandas etc.)

Repetition: Running using Docker

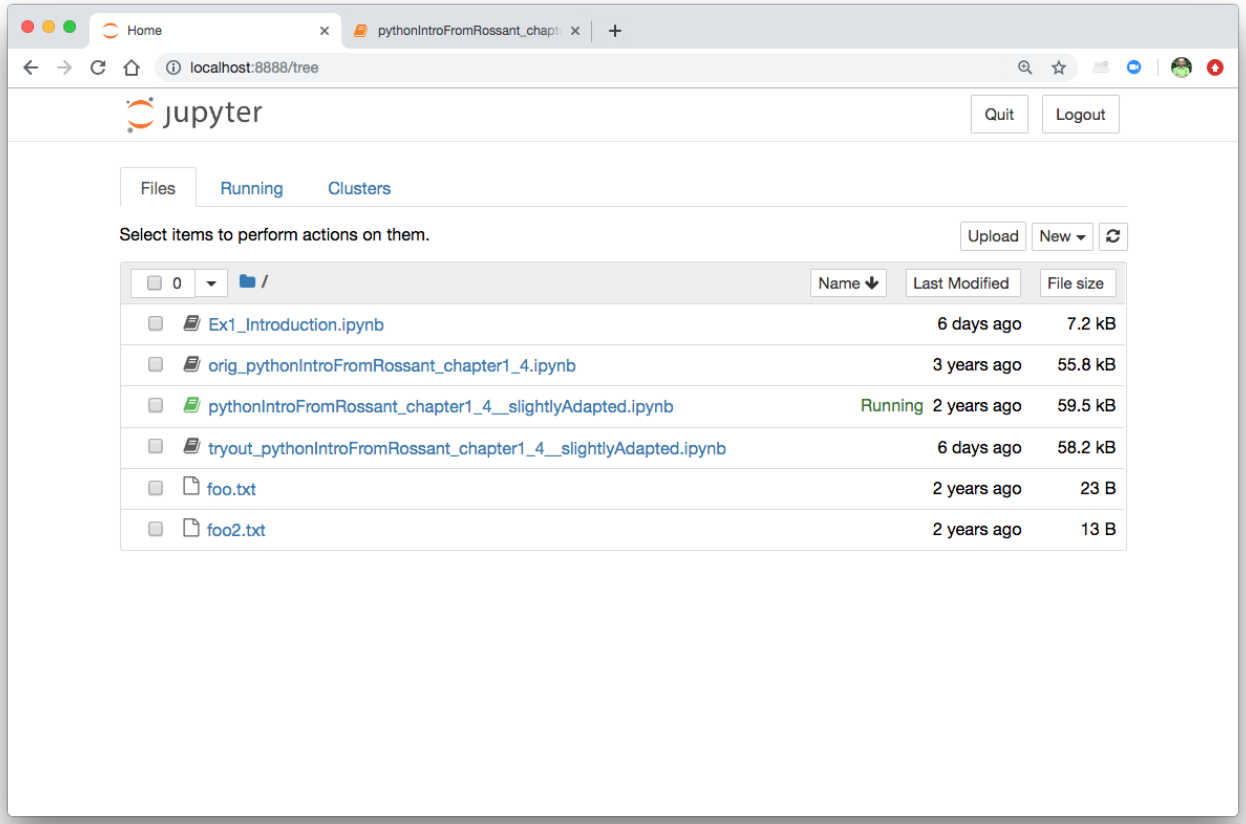
- in order to be able to modify and later submit the .ipynb exercises, you need to mount your working directory
- execute the following command in Terminal/Command Line:

```
docker run -it -p 8888:8888 -v
  your/path/to/working/space/
:/home/jupyter arbitraryimagename
```

- after running this, you should get an URL
- copy-paste it into your preferred browser and you're ready to go!

Repetition: Running using Docker

- using the URL from the output in a browser on your system you get something like this:



- check Moodle for a step-by-step installation guide and troubleshooting

Exercise Sheet 1: Introduction to Python

- goal: get used to working with Jupyter Notebook and Python
- warm-up ex.: use loops to create a simple #-Pyramid:

- in the main ex. you will learn how to:
  - work with large datasets
  - choose the right format for your variables
  - use powerful tools to create, manipulate and display graphs

The Data: The Simpsons Characters

- *nodes.csv*
  - each vertex represents a character
- edges.csv
  - edges between the source character and target character
  - an undirected graph showing the characters which appeared together in an episode
- *ep-char.csv*
  - shows which character has appeared in which episode

Task 1.1: Python Pyramid

a) Create a function which takes a number of levels (N) and prints a pyramid that looks like this for N = 4:

```
   #
  ###
 #####
#######
```

Hints:

- Do not forget the spaces left and right of the pyramid, except on the ground floor. In the example above 3 spaces to the left and right at the peak.
- In order to execute a code cell, press Shift + Enter.

```
def printPyramid(N):
     # TODO your code here

printPyramid(4)
```

b) Extend the program by implementing user input. The user is asked to enter a number for the levels of the pyramid. Afterwards the pyramid is printed.

```
# TODO
num_levels =
printPyramid_sol(num_levels)
```

Tasks (cont.)

Task 1.2 : The Simpsons are introducing Social Computing

a) Your first task is to **drop unwanted rows** in the episodes dataframe. We are only interested in Seasons 1-10.

Hint: Unwanted rows are rows which have an `episode_id` higher than `HIGHEST_EPISODE`. **Note:** Please note that this operation will only delete the rows without changing the weights of the characters. Do not worry about this.

In [ ]:
```
# TODO: drop rows of the df_epchar DataFrame.



# Delete these row indices from the dataframe
df_epchar.drop(indexNames, inplace=True)
```

b) Now you can **merge** the DataFrames together to link the required information. This is not unlike the join operation in SQL.

Since we are only interested in characters from the first 10 seasons, create a DataFrame `df_merged` which only contains characters from the first 226 episodes.

In [ ]:
```
# TODO:
df_merged =


# df_merged now consists of the characters which appear only in the first 10 seasons
df_merged.drop(['episode_id', 'character_id'], axis=1, inplace=True)

# TODO: now we have unnecessary information, drop the duplicates.


df_merged
```

Tasks (cont.)

c) Now use the DataFrame of limited characters and **merge** them with the edges

In [ ]:
```
# TODO:
# Hint: Use a left join, left_on='Id', right_on='Source'
df_merged2 =


# Drop Type, as it is not that interesting
df_merged2 = df_merged2.drop(['Type'], axis=1)
df_merged2
```

d) Now we are only interested in **characters who have appeared at least 20 times together. Select those.**

In [ ]:
```
# TODO: drop rows of the df_episodes DataFrame.


# Delete these row indices from the dataframe
df_merged2.drop(indexNames, inplace=True)
df_merged2
```

Tasks (cont.)

e) Now you have to include your alter ego into the network. Create a pandas Series with your name, your Id (which is 1337) and weights. Connect yourself to Homer Simpson.

```
# TODO:
# Create a series for your character who is connected to homer 234 times
# and add it to the dataframe


# TODO: append the list of series to the pandas data frame

# Create the graph from the dataframe
graph = nx.from_pandas_edgelist(df_merged2, source="Id", target="Target", edge_attr=True)
```

Tasks (cont.)

f) **Draw** the resulting graph with the given options. Choose 2 layout [6] options that seem the most suitable for the data. Briefly discuss why you chose these over the others.

In [ ]:

```python
# Relabel the graph
df_nodes_labels_dict = df_nodes.set_index('Id').to_dict()['charname']
graph = nx.relabel_nodes(graph, df_nodes_labels_dict)

# Set the edge color according to the weight
edges,weights = zip(*nx.get_edge_attributes(graph,'Weight').items())

# Style the graph
options = {
    "font_size" : 14,
    "font_color" : '#552222',
    "node_color" : '#22FF22',
    "width" : 5.0,
    "edgelist" : edges,
    "edge_color" : weights,
    "edge_cmap" : plt.cm.Blues
}

plt.figure(1,figsize=(40,40))

# TODO: plot the graph


```

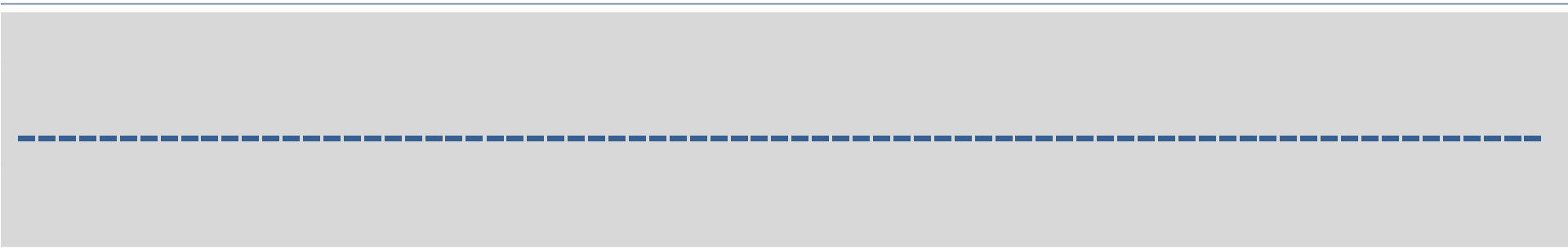
TODO: Write your observations here:

Tips and tricks

- for most TODOs it is sufficient to look at the pandas manual and use pandas library functions
- You can get a better overview of the dataframe by printing it

Submitting your solution

- work by expanding the .ipynb iPython notebook for the exercise that you downloaded from Moodle
- save your expanded .ipynb iPython notebook in your working directory
- submit your .ipynb iPython notebook via Moodle (nothing else)
- remember: working in groups is not permitted. Each student must submit their own .ipynb notebook!
- we check for plagiarism. Each detected case will be graded with 5.0 for the whole exercise
- deadline: check Moodle



Citations

(1) [Beazley 2013) David Beazley: Python Essential Reference, Safari Books 2013, E-Book available via www.ub.tum.de

(2) [Rossant 2015] Learning IPython for Interactive Computing and Data Visualization (SECOND EDITION) by Cyrille Rossant, 175 pages Packt Publishing, October 2015