University of Innsbruck – Department of Computer Science

Methods of Software- und Data Engineering

Prof. Dr. Michael Felderer, Yakup Ipek

EXERCISE SHEET 1

DEADLINE:

17 October 2018 - 3 p.m.

TOPICS:

Jupyter Notebook, Python, Github API, Pandas, Numpy

DESCRIPTION:

The following exercises allow you to gain hands-on experience with some python libraries that are widely used within the domain of Data Science.

SUBMISSION:

Please upload your solution in OLAT according to the following instructions:

Solve the tasks within either a single or multiple notebook files. Add explanations to your solutions using the Markdown cell. Feel free to use any feature provided by Jupyter Notebook to enhance your solution. Name your solution files according to the following designation rules:

Notebook-File:

When submitting your solution please adhere to the following structure: <Surname>_ExSheet_<Sheet Nr.>_<Task Nrs. Separated with a '–'>.<file-ending>; e.g. Ipek_ExSheet_1_1.ipynb, Ipek_ExSheet_1_1-2-3.ipynb

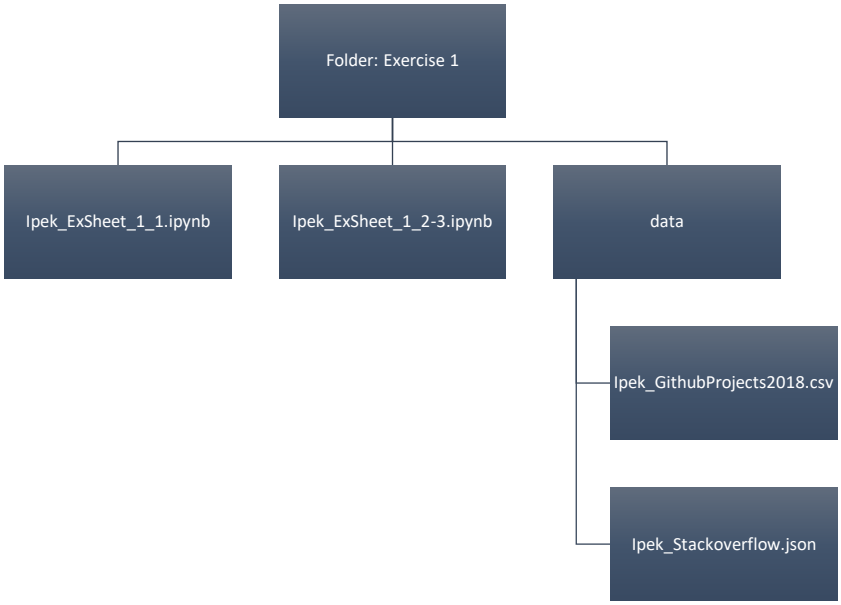
Data (CSV, JSON, etc.):

Any data source on your local filesystem that is explicitly accessed within your scripts should be placed within a **data** folder. The naming convention follows a slightly different structure than above: <Surname>_<Desired designation>.<file ending>

E.g. Ipek_GithubProjects2018.csv or Ipek_GithubProjects2018.json

PLEASE SUBMIT YOUR SOLUTION AS A SINGLE ZIPPED FILE AND NAME IT ACCORDINGLY:

<Surname>_ExSheet_<Sheet number>.zip/tar.gz; e.g. Ipek_ExSheet_1.zip or Ipek_ExSheet_1.tar.gz



LINKS:

Github API	https://pygithub.readthedocs.io/en/latest/apis.html
Pandas	https://pandas.pydata.org/pandas-docs/stable/tutorials.html
Package installation guide Note: Python packages can also be installed within a Virtual Environment	Global installation: https://docs.python.org/3/installing/index.html Virtual Environment: https://packaging.python.org/tutorials/installing-packages/#creating-virtual-environments
Jupyter Notebook Shortcuts	https://www.dataquest.io/blog/jupyter-notebook-tips-tricks-shortcuts/

PREREQUISITES:

In this exercise sheet the following python packages are required and need to be installed:

- jupyter
- numpy
- pandas
- pygithub

Task 1: Jupyter Notebook & Python 2P

1. Create a jupyter notebook and enter the Leibniz formula into your document using a markdown cell which supports LaTeX. Compute $\boldsymbol{\pi}$ with the Leibniz formula using maximally 5 lines of code (be creative 😊)

$$1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \ldots = \frac{\pi}{4}$$

Tipp: leverage built-in functions, lambda expressions, list/dict comprehensions, libraries

Task 2: Numpy Array vs Lists 2P

In this task you will examine the differences between ***lists*** and numpy ***arrays***.

1. Conduct a benchmarking where you measure 1 million consecutive data accesses (within loop)
2. Measure the execution time of the accesses using the ***time*** package
3. Repeat the measurement 10 times and compute finally the average execution time
4. Explain the disparity in the performance.
5. State reasonable use cases for each of the data types (list, numpy arrays)

Task 3: Pandas 3P

Required: wowbgs2.csv (data set about World of Warcraft; can be found on OLAT)

1. Read the wowbgs2.csv data into a dataframe.
2. Indicate which columns contain ***Nan*** values
3. Indicate the number of indices (rows) after dropping any rows that contain ***Nan*** values
4. Indicate the number of columns after dropping any columns that contain ***Nan*** values
5. Sort your data set by the attribute ***Class***
6. List the number of entries for each ***Faction***
7. Which class of the fractions scored the most wins?
8. State in which case it would make sense to keep ***Nan*** cells. What is a good value to fill ***Nan*** cells?

Task 4 : Repository Mining 3P

In task 4 you will get hands-on experience with the Github-API. Below you can find a code-snippet demonstrating the usage of the API. For more examples please check out the pygithub reference page.

```python
from github import Github

user = Github('******************') #(optional) enter here your access token; note
that authorized users' rate limit for API calls is higher

def print_repo_info(repository):
    print(repository.full_name,repository.language)

def search_for_repos_and_print(query, sort, order):
    repos = user.search_repositories(query,sort,order)
    for repo in repos:
        print_repo_info(repo)

if __name__ == "__main__":
    search_for_repos_and_print("tetris","stars","desc")
```

1. Use the Github API to retrieve information of all repositories according to the following fields:

search keyword	sort	order	created
gameoflife	stars	desc	2018

Note: There is a rate limit for the searching API which limits the number of requests per minute and the results for each search are narrowed down to 1,000.

As this query exceeds 1,000 items you need to find a way to get all repositories.

2. Save only the attributes shown in the table below in json-format into a file and place it into the **data** folder.

fields
name
language
created_at
forks
size
watchers
url