Universität Hamburg
Department Informatik
Wissenschaftliches Rechnen
Dr. J. Kunkel, H. Hatzel

Exercise Sheet 1
Big Data Analytics
WiSe 2016/2017
Due Date: 2016-10-28, 480 P

This exercise will introduce you to the basics of Python and R. Those are the two programming languages that we will be using for the rest of the semester.
Each task has an associated amount of points which should represent the time you work on the exercise in minutes. On each of the sheets you will be required to reach at least 50% of the total score. Results must be submitted in groups of two students each.

We have a system for submitting your results:
`http://wr.informatik.uni-hamburg.de/abgabe/bd-1617`. An ID will be sent to your group members by email, everybody having this shared key can access and submit the results. If the groups change, simply re-register with the new group members and you will receive a new ID. Please don't submit binaries created by your compiler, just the source code and other documents.

Feedback regarding lecture and exercise can be submitted there as well. Please document how much time each of the tasks took you to complete. We use these to evaluate the quality of our tasks and help identify what the problems with submissions might have been.

Please document your code, without sufficent documentation you won't recive any points. Should any problems arise please don't hesitate to contact the mailing list:

`bd-1617@wr.informatik.uni-hamburg.de`

1 Cluster Account (30 P)

In this task, you will learn how to access the cluster and using the shell.
Please login with your cluster account and make yourself familiar with the essential Linux commands. Information on cluster usage is available on our Wiki:

`http://wr.informatik.uni-hamburg.de/teaching/ressourcen/start`

Take a look at the "Beginners Guide"! The following tasks are to be completed:

1. If you have a Windows system: install Putty or a comparable SSH-Client (see the Beginners' Guide)
2. *Login* on the cluster using the user name and password that were provided by email. **Use the `ssh` command to connect to `abu1`, software needed for future tasks of this lecture will typically only be installed on `abu1`.**
3. *Navigation using the CLI (Command Line Interface)*
   a) Familiarize yourself with the usage of man-pages: `$> man man`
   b) Print your current working directory: `$> man pwd`
   c) Show all files in your home directory: `$> man ls`
   d) Create a directory named `testdir`: `$> man mkdir`
   e) Change your working directory to the newly created one: `$> cd testdir`
   f) Output the current working directory once more.
   g) Create an empty file named `testfile`: `$> man touch`

h) Rename the file to `testfile2`: `$> man mv`

i) Copy the renamed file to `testfile3`: `$> man cp`

j) Delete the file `testfile2`: `$> man rm`

Question: Why is the manual page `cd` titled ”POSIX Programmer’s Manual” whereas other manual pages have the title ”User Commands” ? (hint: Study the `cd` manual-page in detail.)

Submission:

`1-cluster-navigation.txt`	Answers to the questions and documentation of the commands you executed. Feel free to just copy&paste the latter from your shell.

2 Hello World in Python (Python & R) (150 P)

This task’s goal is to setup your development environment and perform first steps in each of the languages. We will use RStudio for R and PyCharm for Python.

2.1 Python

We recommend that you use the PyCharm IDE for development, the free version comes with all required functionality. If you prefer development in another text-editor or IDE, please feel free to use it. PyCharm is available for all major platforms on the official website:
`https://www.jetbrains.com/pycharm/download/`
If you have no Python experience please consult this introduction:
`https://developers.google.com/edu/python/`

We will use NumPy to perform mathematic computation. Take a look at the NumPy introduction:
`http://wiki.scipy.org/Tentative_NumPy_Tutorial`.
Complete the following tasks:

- Write a program that prints the text "Hello World“.
- Execute the program.
- When executing from the shell, how can you execute the script without expressly calling the Python interpreter? (Document your answer in a comment inside your program)

2.2 R

An R Introduction is available at: `http://data.princeton.edu/R/introducingR.pdf` and using the package "swirl“ (see the lecture slides for details).

Complete the following tasks:

- Use `swirl` for an interactive introduction to R.
- Write a program that prints ”Hello World”.
- Execute the program from the interactive shell.
- Execute the program from the command line using Rscript.

If you research R on the Internet you will notice two different ways that assignment can be done. The `<-` as well as the `=` operator. We recommend using the `=` operator in our exercises to prevent confusion with the expression $x < -y$.

Submission:

`2-hello.`$(py|R)$ Your source code for each of the languages

3 Reading CSV Data (Python & R) (180 P)

A popular, portable, file format for structured data is CSV. In this task, you will read a CSV file and print the mean of one of it's columns. Your program should potentially work with any CSV file (conforming to certain standards). The user should be able to specify which columns mean is printed from the command line. Your program in Python and R respectively should be run like this:

```
python csv.py [FILE] [COLUMN_NUMBER]
Rscript csv.R [FILE] [COLUMN_NUMBER]
```

For testing purposes we have provided a CSV files with measurement series on the cluster:
`/home/bigdata/1/pitbull-ddw1024k.csv`

Complete the following tasks:

- Compute and print the mean of the `deltatime` column using your program.
- Use the Unix-tool `time` to evaluate the performance of both your programs. Which one does perform better?

3.1 Hints and Code-Skeleton

You *may* use the following structure and functions (feel free to use any other function from the standard libraries if they help you, but don't use functions that parse the CSV for you).

3.1.1 Python

```python
import sys

# Commandline arguments are avalible in the sys.argv list
print(sys.argv)

# Implement this function, do not use a function to completely parse the CSV file!
def computeMeanCSV(filename, column_number):
  ...
  return mean
```

3.1.2 R

```r
# The commandline arguments are available in commandArgs
args = commandArgs(trailingOnly = TRUE)

# Implement this function, do not use a function to completely parse the CSV file!
computeMeanCSV = function(dateiname, spaltenNr){
  ...
}
```

Submission:

`3-csv.`$(py|R)$ Your source code
`3-csv-speed.txt` Your runtime measurements with a short evaluation.

4 Word Frequencies in Moby-Dick (120 P)

In this task, you will write a Python program to compute word frequencies for text. We will work with the book "Moby-Dick" by Herman Melville.
You can find it as a text file on the cluster at: `/home/bigdata/1/moby-dick.txt`

4.1 Visualization of Individual Word Frequencies

Your program should take a word as input via command line argument, then split the book into its chapters and for each chapter calculate the relative frequency of the word given by the command line.
Your program will compute the occurrences of any given term across the books chapters. Visualize the frequency in a graph (see the hints).

4.2 Aggregation of Word Occurrences

Now we want to output the absolute occurrences of *each* word in each chapter. Program output should be a CSV file with the following header:
`chapter-name, chapter-length, word-occurrences`
The word-occurrences should be stored as JSON.

4.3 Most Common Words

Write a program to print the 10 most common words in the whole book.

4.4 Hints and Code-Skeleton

Visualization of the word frequency can be done modifying the code provided below:

```
#!/usr/bin/env python3
import matplotlib.pyplot as plt

if __name__ == "__main__":
    numbers = [1,22,33,5,44] # Save your samples here
    plt.bar(range(len(numbers)), numbers, 0.35)
    plt.ylabel("selected word")
    plt.xlabel("occurrence over all chapters")
    plt.show()
```

For this task the module *matplotlib* is required, it is already installed on the cluster. In case you want to test your work locally, install the package via the command `pip` or your distribution's package manager.
You may use dictionaries and the `json` module.

Submission:

`4-book-graph.py`	Your source code to visualize a word's occurrence.
`4-book-csv.py`	Your source code to extract the word occurrences.
`4-book-most-common.py`	Your source code to find the most common words.