Assignment no. 1

Introduction to Python

Scenario: We received files containing data from several experiments conducted by an external group. They provided us with several files of different types containing the data. For a first analysis, we want to create a look at this data. The parts we are interested in are stored in files ending with `.exp1.data` after a specific header. The number of the experiment can also be taken from the header. We now want to create a histogram of the column `height` of all files and a plot of how the average `age` values per experiment develop from experiment to experiment.

To keep our program modular, we will use functions for different subtasks of our analysis.

Exercise 1 (14 points) Write a Python function that scans a directory and included subdirectories for files ending with `.exp1.data`. Print the number of found files and the names of the files to the command line.

The function should take one argument `directory`, which is the directory that should be scanned. The function should return a sorted list of the full filenames (including the paths) of all found files. Store the function in file `ex1.py`.

Hint: The glob module can help you to scan a directory and its subdirectories.

Exercise 2 (14 points) Write a Python function that reads in the content of a text file (in text mode) and returns the values in specified columns. For this it should take the arguments `filename` which is a string indicating the filename and `columns`, which is a list of strings indicating the desired column names. It should then look for the header

```
# Header Start
# Experiment: <experimentnumber>
# Columns: <columnname1> <columnname2> <columnname3>
# Data start
```

which can occur at any line in the file. Instead of `<columnname>` the name of the column is written. There can be multiple column names, separated by any number of white spaces. The column names should be treated case insensitive.

If the file contains a valid header, your function should check if the desired column names are existing columns. If no valid header can be found or the two column names do not exist within the header, your function should raise AttributeErrors with respective error messages.

Otherwise, your function should read the values in the columns with the desired column names. Your function should then store these values in a list or an array for all lines where all desired column values exist. If no lines where all desired values exist can be found, your function should raise a ValueError.

The line

```
# Data end
```

will indicate the end of the data in the file. Read the values only until this line. If this line does not exists, raise an AttributeError with an error message.

Check if the values are numbers by trying to convert them to floats. If this is not possible, ignore the values in this line.

Finally, your function should return the experiment number and the values read from the desired columns as dictionary. The return statement should look something like this:

```
return experimentnumber, values
```

where

```
values=dict(columname1=value_list1, columname2=value_list2, ..)
```

Store the function in file ex2.py.

Hint: There are multiple ways to solve this exercise, the re or csv modules might be helpful.

Exercise 3 (14 points) Write a Python program that imports the functions from exercise 1 and 2. The program should take 1 argument (the directory to search for files in) as command line argument.

With the help from the functions from exercise 1 and 2, scan the directory for files ending with .exp1.data. Print no files found! if the directory does not contain the searched-for files. Otherwise, apply the function from exercise 2 on the list of filenames and extract the columns age and height from all files. If this function raises an AttributeError or ValueError exception, catch it, print Invalid file <filename> found but ignored!, and continue with the program.

Hint: Store all height values for all experiments in one list and store the mean over the age values per experiment in a dictionary with the experiment number as key and the mean age value as value.

Create an output directory with the name <directory>_processed, where <directory> is the directory name specified by the command line argument.

Plot the average age value per experiment in a simple line plot and save it to the file average_age.png in the output directory. Use the title average ages for the plot and age and experiment for the axis titles of the plot. The experiments should be sorted by experiment number.

Finally, store all height values (across all experiments) in a numpy array with datatype np.float32 and save it via pickle to heights.pkl.

If everything went well, print Ex 3 is done! to the command line.

Hint: You can use the following code to create the directory if it does not exist already:

```
import pathlib
pathlib.Path('/my/directory').mkdir(parents=True, exist_ok=True)
```

Exercise 4 (14 points) We will now use Python as sort-of shell-script alternative:

The program should take 1 argument (the directory to search for files in) as command line argument.

Use the `subprocess` module to start the program from exercise 3 with the directory to search for files in as command line argument.

Wait for the program to finish and then load the values from the `heights.pkl` pickle file.

Create a histogram of the `height` values in the pickle file and save it to the file `heights.png` in the output directory `<directory>_processed`, where `<directory>` is the directory name specified by the command line argument. Use the title `heights` for the plot.

If everything went well, print `Ex 4 is done!` to the command line.

Submission: electronically via Moodle:

> `https://moodle.jku.at/`

Deadline: December 12th, 2017, 13:00.
Follow the Instructions for submitting homework stated on the Moodle page!