Cardiff School of Computer Science and Informatics

Coursework Assessment Pro-forma

Module Code:	CMT114
Module Title:	Python for Data Analysis
Lecturer:	Dr. Matthias Treder, Dr. Luis Espinosa-Anke
Assessment Title:	CMT114 Coursework
Assessment Number:	1
Date set:	25-10-2019
Submission date and time:	22-11-2019 at 9:30 am
Return date:	

This assignment is worth 40% of the total marks available for this module. If coursework is submitted late (and where there are no extenuating circumstances):

1 If the assessment is submitted no later than 24 hours after the deadline, the mark for the assessment will be capped at the minimum pass mark;

2 If the assessment is submitted more than 24 hours after the deadline, a mark of 0 will be given for the assessment.

Your submission must include the official Coursework Submission Cover sheet, which can be found here:

https://docs.cs.cf.ac.uk/downloads/coursework/Coversheet.pdf

Submission Instructions

Your coursework should be submitted via Learning Central by the above deadline. You have to upload the following files:

Description		Type	Name
Cover sheet	**Compulsory**	One PDF (.pdf) file	Student_number.pdf
Your solution to question 1	**Compulsory**	One Python (.py) file	Q1.py
Your solution to question 2	**Compulsory**	One Python (.py) file	Q2.py
Your solution to question 3	**Compulsory**	One Python (.py) file	Q3.py

Replace 'Student_number' by your student number, e.g. C1234567890. Make sure to include your student number as a comment in all of the Python files!

For question 1 and 2 submission follow below instructions:

Download the following files from Learning Central:

- Q1.py
- Q2.py

- Q3.py
- IEEEexample.docx
- APAexample.docx
- nobelprizes.json
- whitelist.txt

Test your implementation:

For Q1 and Q2, you can execute the function from the command line

```
> python3 Q1.py

> python3 Q2.py
```

You can change the parameters of the function call in the main part of Q1.py and Q2.py. For Q3, you can use

```
> python3 Q3.py my_paper.docx IEEE
```

where `my_paper.docx` is an example journal paper and IEEE represents the target style.

Any deviation from the submission instructions above (including the number and types of files submitted) will result in a mark of zero for the assessment or question part.

Staff reserve the right to invite students to a meeting to discuss coursework submissions

---

Assignment

Answer all of the following questions.

Question 1 – Random converter (Total 30 Marks)

Implement a function `random_converter(x)` that takes a variable x. It then returns the value of x that has been randomly converted into *int*, *float*, *bool*, *string* or *complex*.
For instance, for `x = 12` (an integer) `random_converter(x)` can return '12' (a string) or 12.0 (a float).

Further instructions:

- x can be any type in *int, float, bool, string* or *complex.*
- the assignment needs to be truly random, that is, if repeated several times, different outcomes should result.
- If x cannot be converted (e.g. the string "house" cannot be converted to a number) the function should print "cannot be converted" and return *none*.

As a starting point, use **Q1.py** from Learning Central. Do not rename the file or the function.

Question 2 – Nobel Prize Data Mining (Total 35 Marks)

You are provided with a dataset in json format (*nobelprizes.json*), which contains information about Nobel prize winners. Specifically, you will find information about a winner's name, category, reason for award, year, etc. To load the dataset, you will need to use the json module (`import json`), and the `d = json.load(file_object)` method.

Question 2.1

- Implement a function `report()`, which takes as input the json file loaded as a Python **dictionary** (which is the default data structure returned by the `json.load()` method). This function should return **a Pandas DataFrame**, where you include the **years** and **categories** in which a Nobel Prize was awarded and those in which it was not. You are not expected to infer any missing information, you should only include years and categories for which there is an explicit entry in the original dataset. The result should be of the following form (made up values):

year	**category**	**awarded_or_not**
1963	chemistry	True
1976	physics	False

Further instructions:

- There is no field called 'awarded_or_not' in the dataset, you have to find this information elsewhere. Discuss your solution in the code as comments.
- Years should be represented as integers, categories as strings and awarded_or_not values should be boolean.
- Column names should be 'year', 'category' and 'awarded_or_not'.

Question 2.2

- Write a function `get_laureates_and_motivation()` which takes as input three arguments: the nobel prize dictionary (same as in Q2.1), year (a string) and category (a string). This function returns **a Pandas DataFrame** containing one row per laureate (i.e., a person who has won the Nobel prize). The returned DataFrame should be of the form below (made up values):

category	**year**	**id**	**laureate**	**motivation**	**overall_motivation**
chemistry	1963	501	john doe	he was great	he was among great minds
chemistry	1963	700	susan sarandon	she was great	NaN

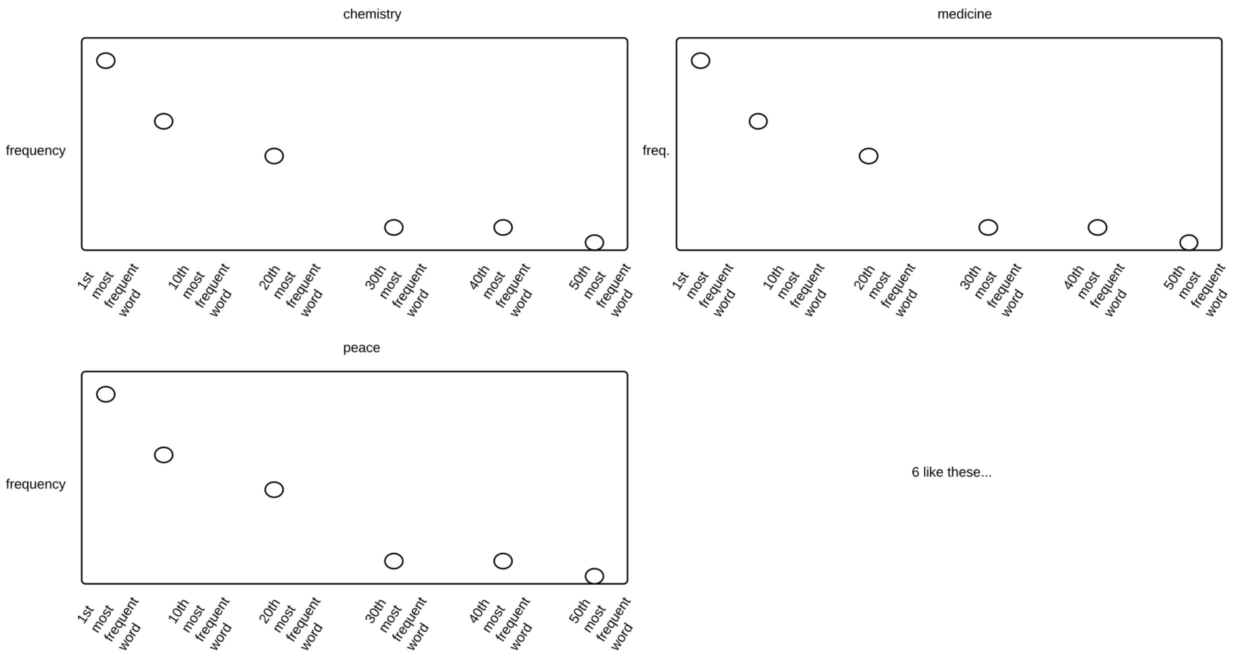
Further instructions:

- The *id* values refer to the laureate id as per their identifier in the original dataset.
- Overall motivations are reasons for awarding Nobel prizes which apply to more than one person in the same batch. However, not all laureates have an overall motivation associated. In those cases, you should insert a *NaN* value in their 'overall_motivation' field.
- Categories, laureates and motivation should be *strings*, years and ids should be *integers*, and overall_motivation should be either *string* or *NaN*.

- Use the column names shown in the sample table above, do not change them.

Question 2.3

- Write a function `plot_freqs()` which generates six plots, one for each category. The x-axis should contain the 1st, 10th, 20th, 30th, 40th and 50th most frequent word across the motivation sections for each category. The y-axis should refer to the frequency of each word in that category. The resulting plot should have a similar arrangement as the one below.



Further instructions.

- You should only count the words provided in *whitelist.txt*, a text file available in learning central, with one word per line. Do not count others.
- Your figures should have a title, legend, the frequency of each word, tick marks, labels in the x axis for each word, be readable (e.g., big enough fonts), etc.

As a starting point, use **Q2.py** from Learning Central. Do not rename the file or the functions.

Question 3 – Citation Style Manager (Total 35 marks)

In scientific publications, a reference to a previous work (source) that is discussed in the manuscript is called a *citation*. In different scientific disciplines, and sometimes even different journals, different so-called *citation styles* are used. The citation style defines how a citation is formatted. We will consider two different citation styles in this question:

- **APA style**: citation style of the American Psychological Association (https://www.mendeley.com/guides/apa-citation-guide), see also Wikipedia page (https://en.wikipedia.org/wiki/APA_style). This style is widely used in Psychology and Social Sciences.
- **IEEE**: citation style of the Institute for Electrical and Electronics Engineers (IEEE) is used in IEEE journals which cover engineering and related disciplines (https://pitt.libguides.com/citationhelp/ieee). See the Learning Materials/Coursework folder on Learning Central for more information on the IEEE style.

There are two main aspects to a publication where citation styles apply:

1. *In-text citations*: These are used in the text body whenever one refers to, summarises, paraphrases, or quotes from another source. This is an example from Wikipedia (https://en.wikipedia.org/wiki/APA_style) for a sentence including an in-text citation of a paper by Schmidt and Oh in APA format:

   > *In our postfactual era, many members of the public fear that the findings of science are not real (Schmidt & Oh, 2016).*

   In IEEE format, references are given as numbers in square brackets. Example:

   > *This is compounded by the fact that the field is evolving from work performed by an individual that does data science to a team that does data science [1].*

2. *Reference list*: In a scientific publication, the last section is typically the References section, which provides full details on the in-text citations. For instance, the full reference corresponding to the Schmidt & Oh (2016) in-text citation above would be:

   > *Schmidt, F. L., & Oh, I.-S. (2016). The crisis of confidence in research findings in psychology: Is lack of replication the real problem? Or is it something else? Archives of Scientific Psychology, 4(1), 32–37. https://doi.org/10.1037/arc0000029*

   In an article using IEEE format, every reference in the reference list needs to be numbered:

   > ***1.** J. Saltz, "The Need for New Processes Methodologies and Tools to Support Big Data Teams and Improve Big Data Project Effectiveness", Big Data Conference, 2015.*

Your task: Implement a function `change_style(filepath, style)`, which takes as input two arguments: (1) `filepath`, which can be either IEEEexample.docx or APAexample.docx and (2) `style` (a string being either IEEE or APA), and swaps their citation style (i.e., converts IEEE citations into APA and vice versa). **You are not expected to consider cases outside the two documents provided.**

Detail instructions:

- To ease the task, you will be working with .docx files (working with PDFs or online sources would be more difficult). Two example files (IEEEexample.docx and APAexample.docx) are provided in Learning Central.
- Use the `python-docx` package to read, manipulate, and save doc files. You can install it using e.g. `pip install python-docx`. Check the webpage (https://python-docx.readthedocs.io/en/latest/index.html#) or other online sources to familiarize yourself with the package.
- After conversion, save the file by appending '_APA_style' or '_IEEE_style' to the filename (e.g. 'myfile_IEEE_style.docx').

- We make the following simplifications
    - In the *reference list*, you **do not need** to change the formatting of individual references. Only make sure that there is numbering (for IEEE style) as opposed to no numbering (for APA).
    - For APA, the reference list should be sorted alphabetically. Example:
      - Barnes (2010) …
      - Churchill (2015) …
      - ...
    - For IEEE, the reference list should be sorted numerically (smaller to greater), where 1 refers to the first in-text citation in the paper, 2 refers to the next citation, and so on. Example:
      - [1] Zhang and Wu. …
      - [2] Antonidas. …
      - ...
    - Your program should re-format all *in-text* citations.

- To implement your programme, you should only use basic Python including string operations, as well as the docx module. Usage of Numpy, Pandas, the regular expression module `re`, or any other modules not used in the first 4 lectures is not permitted!

As a starting point, use **Q3.py** from Learning Central. Do not rename the file or the function.

Learning Outcomes Assessed

- Using the Python programming language to complete programming tasks
- Familiarity with basic programming concepts and data structures
- Reading and writing files

Criteria for assessment

Credit will be awarded against the following criteria; the coursework will allow students to demonstrate their knowledge and practical skills and to apply the principles taught in lectures. The functions you have implemented will be tested against different data sets. The score each implemented function receives is judged by its functionality, efficiency, and/or quality. The below tables explain the specific criteria for each question.

Criteria	Distinction (70-100%)	Merit (60-69%)	Pass (50-59%)	Fail (0-50%)
Q1	Excellent working condition with no errors	Mostly correct. Minor errors in output	Major problem. Errors in output	Mostly wrong or hardly implemented

	Criteria	Distinction (70-100%)	Merit (60-69%)	Pass (50-59%)	Fail (0-50%)
Q2	Functionality (70%)	fully working application that demonstrates an excellent understanding of the assignment problem using relevant python approach.	All required functionality is met, and the application are working probably with some minors' errors	Some of the functionality developed with and incorrect output major errors.	Faulty application with wrong implementation and wrong output
	Quality (30%)	Figures are elegant and show an excellent understanding of visualisation principles including tick marks, labels, colouring, and titles.	Figures show a good understanding of visualisation principles.	Figures show a basic understanding of visualisation principles.	Missing figures.

	Criteria	Distinction (70-100%)	Merit (60-69%)	Pass (50-59%)	Fail (0-50%)
Q3	Functionality (70%)	fully working application that demonstrates an excellent understanding of the assignment problem using relevant python approach.	All required functionality is met, and the application are working probably with some minors' errors	Some of the functionality developed with and incorrect output major errors.	Faulty application with wrong implementation and wrong output
	Efficiency (15%)	Excellent performance passing all test cases	Good performance missed some test cases	Passed some test cases with incorrect output.	Did not pass any test case
	Quality (15%)	Excellent documentation with usage of __docstring__ and comments	Good documentation with minor missing of comments.	Fair documentation.	No comments or documentation at all

Feedback and suggestion for future learning

Feedback on your coursework will address the above criteria. Feedback and marks will be returned within 4 weeks of your submission date via Learning Central. In case you require further details you are welcome to schedule a one-to-one meeting. Feedback from this assignment will be useful for next year's version of this module as well as the Computational Data Science module.