Assignments

All asignments will be uploaded here.

1. `clone` (or `pull` from) the **metu-cogs-520** repository to your machine.
2. Work under the directory named in your id.
3. `commit` and `push` the changes.

Conventions:

- all parts under their dedicated folders, each named **part_X** accordingly (e.g. **part_I** for the first part)
- a folder named **in** for your input files
- a folder named **out** for your output files

Try to make your code accessible, meaning taking good care of stucture, naming, comments etc.

Part I

- [**topic**] preprocessing texts
- [**due**] 18th of Nov.
- [**grade**] 10%
- [**programming language**] any
- [**input language**] any (preferably the one you will work with in your project)
- [**tasks**] code a basic preprocesser with a similar scope examplified in the online tutorial, which utilizes third party processes when necessary
    1. review various corpora in your target language and find an appropriate one (preferably noisy)
    2. take a (representative) sample from that corpus. keep it small, no more than 5MB.
    3. preprocess the sample. first **(a)** try to correct spelling by utilizing a 3rd party library (e.g. NLTK for English in Python, İTÜ Turkish NLP pipeline for Turkish etc.). next **(b)** eliminate dates, numbers and punctuation (except those marking sentence boundaries). third **(c)** normalize cases except proper names via a (simple heuristic) process that you will engineer. you can also play with other preprocessing tasks, such as stemming.

Part II

- [**topic**] grammatical tagging
- [**due**] 18th of Nov.
- [**grade**] 10%
- [**programming language**] any
- [**input language**] same as part I
- [**tasks**] code a grammatical tagger that analyzes the output of part I

1. segment input into sentences
2. **(a)** apply dependency parsing to all instances by utilizing a dedicated 3rd party library, **(b)** merge segments conforming to the scope of the original (i.e. input) data points and **(c)** export the output enchanced with the analysis

Part III

- [**topic**] restructuring data
- [**due**] TBA
- [**grade**] 10%
- [**programming language**] any
- [**input language**] same as before
- [**tasks**] code a script that restructures the output of part II
  1. **(a)** read the input and iteratively restructure it so that the resultant data structure is transparent, allowing efficient access considering future data analyses. prefer an embedded process instead of a cascaded one to improve runtimes. **(b)** briefly discuss why you structured the data in the exact way you did.
  2. **(c)** compare xml, json and yaml file formats. choose one that best suits your requirements. briefly discuss your choice. **(d)** export the data in the chosen format.
  3. **(e)** if any, export meta-data that is potentially useful for future analysis in adequate formats. briefly discuss why the additional chunks of data may prove worthy.

Part IV

TBA