Project 2

Submit Assignment

Due Mar 11 by 11:59pm **Points** 100 **Submitting** a file upload **File Types** zip

Code Due: 11:59pm Monday, March 11th as a Git URL, via Canvas

Presentations: Meetings to present your project will take place on **March 12, 14, and 15th** (Tuesday, Thursday, and Friday). You'll sign up for a meeting time in the last week before the project is due. Meeting location is TBA**.**

Project Deliverables:

1. All code must be in Python 3. You can use any Python package or NLP toolkit.
2. You must use a publicly accessible repository such as Github, and commit code regularly. When pair programming, note in the commit message those who were present and involved. We use these logs to verify complaints about AWOL teammates, and to avoid penalizing the entire group for one student's violation of academic integrity. We don't look at the commits unless there's something really wrong with the code, or there's a complaint.
3. Please use the Python standard for imports described here: https://www.python.org/dev/peps/pep-0008/#imports **(https://www.python.org/dev/peps/pep-0008/#imports)**
4. If you use a DB, it must be Mongo DB, and you must provide the code you used to populate your database.
5. Your code must be runnable by the TAs. Thus, your repository must include a readme.txt file that lists the version of the programming language you used, and all dependencies. Any modules that are not part of the standard install of your programming language should be included in this list, along with information on the code repository from which it can be downloaded (e.g. for python, pip or easy_install). If you used code that you instead put in a file in your project's working directory, then a copy of that file should be provided along with the code you wrote; the readme and/or comments in such files should clearly state that the code was not written by your team.

A note on plagiarism: GIT repos from this and previous years are accessible, being public. If you are tempted to look at them, be advised that the code in these repos is not necessarily of great quality. You wouldn't want to copy from a past group who got a poor grade. Also note that we have seen and graded all the code in all these repos rather extensively. If we notice any copying from a past repo, consequences will be severe.

The project:

For your second project, you'll be creating a recipe transformer. Your recipe transformer must complete the following tasks:

1. Accept the URL of a recipe from AllRecipes.com, and programmatically fetch the page.
2. Parse it into the recipe data representation your group designs. Your parser should be able to recognize:
    - Ingredients
        - Ingredient name
        - Quantity
        - Measurement (cup, teaspoon, pinch, etc.)
        - (optional) Descriptor (e.g. fresh, extra-virgin)
        - (optional) Preparation (e.g. finely chopped)
    - Tools – pans, graters, whisks, etc.
    - Methods
        - Primary cooking method (e.g. sauté, broil, boil, poach, etc.)
        - (optional) Other cooking methods used (e.g. chop, grate, stir, shake, mince, crush, squeeze, etc.)
    - Steps – parse the directions into a series of steps that each consist of ingredients, tools, methods, and times
3. Ask the user what kind of transformation they want to do.
    - To and from vegetarian (REQUIRED)
    - To and from healthy (REQUIRED)
    - Style of cuisine (AT LEAST ONE REQUIRED)
    - Additional Style of cuisine (OPTIONAL)
    - DIY to easy (OPTIONAL)
    - Cooking method (from bake to stir fry, for example) (OPTIONAL)

If you come up with your own transformation idea, feel free to ask if it would be an acceptable substitute. We encourage innovation.

4. Transform the recipe along the requested dimension, using your system's internal representation for ingredients, cooking methods, etc.

5. Display the transformed recipe in a human-friendly format.

Your transformations should work on any given recipe. Make sure to test using a wide variety of recipes. Some recipes will be harder than others to transform, and we will use a range of recipe complexity when grading.

Your system can run from the command line and/or from within the Python interpretive environment. Note that some of the things listed above are designated as optional. A group that does a truly fantastic job on all of the steps listed above, but omits the optional items, will probably be in the running for an B+ to B. Optional items will bolster your grade, **although doing all of the optional items and a lousy job on the core items is not recommended.**