Recipe Transformer Assignment Sheet

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

1. (optional) Accept the URL of a recipe from AllRecipes.com, and programmatically fetch the page. If you choose to skip this step, you will be copying and pasting the text or HTML into your program.
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. chopped)
        - (optional) Preparation description (e.g. finely, coarsely)
    - Tools – pans, graters, whisks, etc.
    - Methods
        - Primary cooking method, if one exists (e.g. sauté, broil, boil, poach, freeze etc.)
        - (optional) Other cooking methods used (e.g. chop, grate, stir, shake, mince, crush, squeeze, etc.)
    - (optional) 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. At least three of the following options should be available:
    - To and from two or more of: vegetarian, vegan, pescatarian, lactose-free
    - To a style of cuisine (at least two styles from which the user may choose)
    - To and from two or more types of healthy, such as: low calorie, low fat, low sodium, low carb, low glycemic index
    - From DIY to easy, or from easy to DIY
    - Cooking method (from bake to stir fry, for example; simply replacing one word with another does not count)

    If you come up with your own transformation idea, feel free to ask if it would be an acceptable substitute. We encourage innovation, but innovations that are trivial are less desirable.
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. Bonus points for a point-and-click GUI, though not as many as for an optional goal.

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 a 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.

As with the previous project, your system can run from the command line and/or from within the Python interpretive environment.

With your completed project, you will hand in:

- A table, diagram, or other visual representation of how your group chose to organize the information you parsed from recipes.
- Table(s), diagram(s), and/or other visual representation(s) of your internal data representation for ingredients, tools, methods, and/or transformation dimensions. Note that this is a representation of the knowledge base that your system will be using to transform from your parse of the recipe to the new transformed recipe. A good design here can make all the difference.
- For grading purposes: the code for a program that transforms between your parsing of a recipe and the json representation attached to this document; note that all words should be lower case, all numbers should be in decimal numerical format, and measures that are typically shortened should be expanded (e.g. tsp. → teaspoon). If you did not take on the optional task of identifying ingredient type and preparation, please include those fields but leave them blank. This will be run through an automatic scoring program to assess the accuracy, precision, and flexibility of your recipe parse.
- The code for your system. This should include a readme or comment header 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.

Deadlines

- Tuesday, March 3 5pm – have a group member hand in your two representations at http://goo.gl/forms/xkYqVpvEHR.
- March 12-13 – meet as a group with Larry to demo your project's awesomeness!
- Tuesday, March 17 – have a group member hand in your project http://goo.gl/forms/B8vF97Gyjn.

FAQ

Our recipe representation looks very similar to the output dictionary you illustrate below. Is this okay?

Yes! You are welcome to use that particular structure for your own internal representation, or to do something complete different. Just make sure you can output parsed recipes in the illustrated format for the autograder.

No one on our team knows anything about cooking. How are we supposed to do a good job on this?

It's true that for your system to transform recipes, it will need to know some things about cooking. It may seem impossible to teach a machine something you yourselves don't know, but it can be done! Consider where you might find data that would be of use to your system. Also please keep in mind that if your transformations are lackluster, you can pick up some of the slack by doing more in parsing.

We found this awesome data set and we're going to do this cool machine learning thing to learn the knowledge base!

That sounds like an awful lot of work. Before you go down that rabbit hole, ask yourself the following questions: 1. How long will it take for me to write and run this code, in comparison to other approaches? 2. To what extent will this process improve the results of your parsing and/or transformations?

Make sure it's actually worth your time. If it improves your results, that will have a positive effect on your grade. We might also decide to give some extra credit for this approach, but it would be an amount on par with doing a GUI, not equivalent to an optional item.

Is it okay to hardcode the information in our knowledge base?

Definitely. In past years, some groups have found that data online, but many others have chosen to enter the information by hand.

```
{
    "ingredients": [{
            "name": "salt",
            "quantity": 1,
            "measurement": "pinch",
            "descriptor": "table",
            "preparation": "none",
            "prep-description": "none"
        },
        {
```

```
            "name": "olive oil",
            "quantity": 0.75,
            "measurement": "teaspoon",
            "descriptor": "extra-virgin",
            "preparation": "none",
            "prep-description": "none"

        },
        {
            "name": "parsley",
            "quantity": 1,
            "measurement": "cup",
            "descriptor": "fresh",
            "preparation": "chopped",
            "prep-description": "finely"
        }
    ],
    "primary cooking method": "primary cooking method here",
    "cooking methods": ["chop", "stir", "boil", "simmer", "grate",
"bake"],
    "cooking tools": ["knife", "grater", "dutch oven"],
}
```