Assignment 5: More Scripting - Python Edition

CS3423 - Systems Programming — Rocky Slavin & Sam Silvestro - UTSA

For this assignment, you will use **Python 3** to create a simple templating engine. Your program should take as input a generic template with placeholders for generic data, a set of input files containing data which should be applied to the template, and a date. Instantiated templates using the input data will be output to a subdirectory.

This assignment requires *only* Python. **Do not** use the command line utilities used in the previous assignment. You are encouraged to explore the Python standard library (the `shutil` module may be particularly useful).

Data Format

Data will be stored in the same format as the data in Assignment 1. For each course with an enrollment greater than 50, your program will use the provided template to generate an advisory report specifically for them.

1. Data files (i.e., of the same format as those generated in Assignment 1) will be stored inside a directory specified by the user.
2. Each file within that directory will be named based on the department code (two or three uppercase alphabetic characters), a course number (exactly **four** digits), and ending with an extension of **`.crs`**.
3. A course file consists of *exactly* five lines:
   - `dept_code` (two or three character string) `dept_name` (string with possible whitespace)
   - `course_name` (string with possible whitespace)
   - `course_sched` (string: either `TH` or `MWF`) `course_start` (start with no whitespace) `course_end` (string with no whitespace)
   - `credit_hours` (integer)
   - `num_students` (integer representing number of enrolled students)
4. Example file named **`MAT1311.crs`**

```
MAT Mathematics
Calculus I
TH 8/26/19 12/11/19
3
62
```

Templating

Templates will include variable names to be filled in with data using double square brackets. For any data file of the format described above, each of the variables (including the square brackets) should be substituted with the data's actual value. Your program should work for **arbitrary templates** using the same variables listed below corresponding to the item values described. More than one variable may appear per line.

- `[[dept_code]]`
- `[[dept_name]]`
- `[[course_name]]`
- `[[course_start]]`
- `[[course_end]]`
- `[[credit_hours]]`
- `[[num_students]]`
- `[[course_num]]` (the course number as specified in the filename of the .crs file)
- `[[date]]` (see below)

Example Template:*

```
<html>
  <body>
    <h1>NOTICE OF OVERENROLLMENT - COURSE [[dept_code]] [[course_num]]
        - [[date]]</h1>
    <p>
      Dear Instructor,
    </p>
    <p>
      Your course, [[course_name]], scheduled from [[course_start]] to
          [[course_end]], has exceeded normal capacity with an
         enrollment of [[num_students]] students. This must be
         corrected within thirty days or this issue will be referred
         to the [[dept_name]] department for further review.
    </p>
    <p>
        - Administration
    </p>
  </body>
</html>
```

* This is *only a **single** example of a single template*! You must assume that multiple different templates, with various combinations of variables, will be ran against your program. Experiment with exercising your program using your own custom scripts, as none will be provided to you. This is to emphasize the notion that *no single template should be used as a test instrument; your script will be tested against several unknown-to-you templates*.

Date Argument

The third command-line argument should be a date manually entered by the user of the format **MM/DD/YYYY**. This value should be substituted anywhere where `[[date]]` appears.

Output

All output files should be written to the directory defined by the last argument. This directory may or may not already exist. Each file should be named by the course's department code and number, and with the extension `.warn`.

Script Execution

Your program should be invoked through a single bash file (see below) with **four arguments:** data directory, template file, date, and output directory. Assuming the program executes correctly, no output should be printed to the screen.

```
$ assign5.py ./data assign5.template 12/16/2021 ./output
```

Assignment Data

Sample input files can be found in:
`/usr/local/courses/rslavin/cs3423/Fall19/assign5`.

Script Files

Your program should consist of exactly one file:

- `assign5.py` - the main file which is initially invoked

Extra Credit (5 points)

Allow your program to take *optional* fifth and sixth arguments describing the character(s) surrounding the variables instead of double square brackets. This feature should work for the following characters as either the opening or closing symbol, "/", "|", "}", and "{". Note that these can be in any combination (e.g., starting with "`{`" and ending with "|") If no fifth and sixth arguments are passed, the program should behave as normal. You may assume that if a fifth argument is passed, a sixth will be present, too.

Example:

```
$ assign5.py ./data assign5.template 12/16/2021 ./output '{' '|'
```

The above invocation should replace variables in the template such as `{date|` instead of `[[date]]`.

Extra credit is not given to late assignments. All requirements must be met to qualify for extra credit.

Submission

Turn your assignment in via Blackboard. Your zip file, named **`abc123.zip`** should contain only your Python file.

If you attempt the extra credit, name your file **`abc123_EC.zip`**. Without the **`_EC`**, your submission will be graded as normal.