Assignment 4: More Scripting

CS3423 - Systems Programming — Rocky Slavin - UTSA

For this assignment, you will use **awk**, **sed**, and **bash** 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 sed, awk, and bash. You may use any combination of these programs and are not required to use all of them. **Do not use** Python or any other languages/utilities.

Hint: Since you will need to produce many files from many inputs, it may be useful to use awk to generate sed and/or bash scripts.

Data Format

Data will be stored in the same format as the data in Assignment 1. For each apartment with a balance greater than 0, your program will use the provided template to generate a letter specifically for them.

1. Data files (i.e., 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 apartment number, an integer with exactly **three** digits, followed by the extension `.apt`.
3. An apartment file consists of *exactly* three lines:
   - first_name (string with **no whitespace**) last_name (string with **possible whitespace***)
   - lease_start (string with **no whitespace**) lease_end (string with **no whitespace**)
   - balance (integer)

   * Last names may contain whitespace. You should account for names with *multiple tokens* (e.g., "Bob My Last Name Is Really Long" $\implies$ last_name = "My Last Name Is Really Long")
4. Example file named `323.apt`

```
Ralph Vaughan Williams
2/13/19 2/13/20
900
```

Templating

Templates will include variable names to be filled in with data using double angle brackets. For any data file of the format described above, each of the variables (including the angle 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 above. More than one variable may appear per line.

- **<<first_name>>**
- **<<last_name>>**
- **<<lease_start>>**
- **<<lease_end>>**
- **<<balance>>**
- **<<apt_number>>** (apartment number specified by .apt filename)
- **<<date>>** (see below)

Example Template:

```
<html>
    <body>
        <h1>PAST DUE NOTICE - APT <<apt_number>> - <<date>></h1>
        <p>
            Dear <<first_name>> <<last_name>>,
        </p>
        <p>
            Your rent is past due by $<<balance>> for the apartment in
            which you began residency on <<lease_start>>. If it is not
            paid in full within fourteen days, your contract may be
            terminated before the original end date of <<lease_end>>.
        </p>
        <p>
            -Management
        </p>
    </body>
</html>
```

Date Argument

The third 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 tenant's last name and with the extension **.mail**. For tenants with spaces in their last name, replace the space with an underscore. If the file name already exists (i.e., multiple tenants with the same last name, add an integer to the end of the last name and increment it for each subsequent duplicate name starting at 1. For example, **Vaughan_Williams1.mail** would be the name of the output file for the *second* person with the last name "Vaughan Williams".

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 print to the screen.

```
$ assign4.sh ./data assign4.template 12/16/2021 ./output
```

Assignment Data

Sample input files can be found in:
`/usr/local/courses/rslavin/cs3423/Spring19/assign4`.

Script Files

Your program may consist of multiple files:

- `assign4.sh` - the main file which is initially invoked (required)
- `.awk` files
- `.sed` files

Note: If your program generates any intermediate **awk** or **sed** files during execution, name them beginning with the letter 'g'. Moreover, delete them when your program has completed.

Extra Credit (5 points)

Allow your program to take *optional* fifth and sixth arguments describing the character(s) surrounding the variables instead of double angle 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 passed too.

Example:

```
$ assign4.sh ./data assign4.template 12/16/2021 ./output '{' '|'
```

The above invocation should replace variables in the template such as `{simple_name|` instead of `<<simple_name>>`.

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 bash, awk, and sed files.

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