CSCA20 Exercise 6

Deadline(s)

There will be a pre-grading run on Friday November 9 at 11:59pm. If you submit your work before this deadline, you will receive feedback from the automarker on what grade you would receive on your submission, as well as any errors in your code we've found.

The actual deadline for this exercise will be **Sunday November 11 at 11:59pm**. The latest version of `e6.py` you submit before this deadline (including possibly anything you've submitted before the pre-grading deadline) will be examined to determine your final grade for this exercise. Make sure not to modify the filename, or else the auto-marker won't see your file!

Deadline-related suggestions

I suggest you treat the pre-grading deadline as your actual deadline; ie. try to get all of your work done before then. If you earn a perfect grade, then you're done! And if you happen to make any errors, then you'll have two days to correct them in order to increase your final grade. Not many courses give you this opportunity, so make the best of it.

We will mark the newest version submitted not later than the deadline. Don't submit just once; instead, submit at least once well in advance, so that you know what you're doing, and then keep submitting new versions as you do more of the exercise. That way, if you run out of time on the last function that you just can't do, you'll still get marks for the others.

What you'll be required to do

As usual, you will be completing the bodies of functions in `e6.py`. You will be working with student grade data in a CSV file. Download the file `image_files.csv` and place in the same directory as your `e6.py`.

The format of `image_files.csv` is as follows. The file has no header. The lines of the file are of the format:

```
img_filename,location,date,caption,tags,tags, ...
```

The number of tags will vary line-by-line, so your functions should work for any number of tags. The lines of the body of the file might look like:

```
images/skating.jpg,East York Arena,2014.11.03,Shea
skating.,skating,Shea,boy
```

The file may or may not be empty.

The docstring for each function describes what each function is supposed to do and/or return. It's up to you to implement them to do what their docstrings require. Do *not* modify these docstrings. And also, do *not* modify the format of the functions' arguments or the function names.

Make sure to keep any printing lines inside the `if __name__ == '__main__':` block at the bottom of your code. Specifically, *do not print inside your functions unless specifically instructed to*! Such print statements could mess up the automarker's reports.

You can assume that we will be testing your code using sensible values for the arguments of your functions. Additionally, you can assume that the function argument values will be of the correct type; eg. if the docstring says that an argument is a float, we will only test by giving it floats.

What you *should also* do
You should test your code using the same guidelines from the handouts for the previous exercises as applicable.

Additionally, you should test your code using various versions of `image_files.csv` which you create yourself. For example, vary the number of lines in the file, and vary the number of tags each image has.

For functions that take dictionaries, it's a good idea to try the following:

- Empty dict
- Dict with one key-value pair
- Dict with multiple key-value pairs
- Vary the types of the keys and values (where applicable)

As with before, we won't be grading you on how you test your code, but we will be grading you on how well your code works. And the best way to make sure your code works is to try and break it using a variety of tests.

Attention to detail
Remember that Python cannot make any intelligent corrections for you. It will only do exactly what you tell it to. Any typos or logical errors will make your code behave completely differently from what you want it to! So the best way to make sure you haven't made any of these errors is to test! (See above)

Also, many students have stumbled upon problems with indentation (or lack thereof) leading to syntax errors. Make sure to indent code by exactly four spaces every time you're writing the body of an if statement, for loop, function call, or while loop! If you don't have at least a single indented line of code within the body of one of these, you must use a "pass" statement or else you'll run into a syntax error!

Also, watch those non-English characters! Characters like é or chinese characters in your submission will cause your code to fail to run in the auto-marker, and you'll get a 0.

Style

The adherence of your *entire* modified `e6.py` (even the testing code) to the PEP8 style guide will count towards your grade. After you've implemented and tested your code, run the entire code (not just the function bodies) through the online style-checker provided in the Resources section of the course website and then fix all of the style errors it will no doubt find in your code.

It may seem tedious to write according to the PEP8 style guide when Python will understand non-PEP8-compliant code just as well. But remember that when you're programming, you're programming for humans as well. Writing code which follows style conventions will help humans read it. And as you practice following these conventions, they will become habit and you might eventually find yourself doing it by default.

Thus, the second last thing you should do before you submit is to run your *entire* code through the PEP8 style checker. The last thing you should do is the sanity check (see below).

It is also a good idea to practice following the extra style guidelines from this document:
https://docs.google.com/document/d/1QNoQ1Yt7QLk4vaKrQ8W5G0aDVw4ql6SlQhoQ394fW44/edit?usp=sharing
although we will not be grading for this on this exercise.

Sanity check
If nothing else, make sure your code runs *without any errors* (eg. syntax errors). Furthermore, make sure you can import your module within another file.

In lecture, you've seen how easy it is to forget a bracket and end up with invalid Python code. Even the best programmers make these mistakes regularly. If the code you eventually submit has these kinds of errors, you *will get a 0* on your exercise grade, no matter how well you did everything else! So the last thing you should do before you submit is a final sanity check to make sure you can at least run and import your code without any errors!

What to hand in
Submit your completed `e6.py` on MarkUs
https://markus.utsc.utoronto.ca/csca20f18/?locale=en