# CSCA20 Assignment 2

Due: Wednesday, November 23, 2017, 11:59pm

## 1 Introduction

The purpose of this assignment is to help you gain familiarity with the `dictionary` and `file` types.

In this assignment, you will create a webpage slideshow of images. You do not need to know how to write HTML, that portion will be given to you in a file that you will open and read from. You will add to the file and write it out. To see a sample go here.

## 2 HTML File Structure

HTML files, use *tags* to mark up the page. If you view the above file in a text editor (even Wing) you will see that the file starts with an `<html>` tag and ends with a closing `</html>` tag. HTML tags are of the form open tag `<...>` and close tag `</...>`.

After the `<html>` tag you will notice a `<head>` ... `</head>` tag pair where ... is a bunch of HTML code. You do not have to concern yourself about the code between the `<head>` tags (although if you are interested you can play with the background colours, text colours and image sizes if you wish).

After the header, comes the `<body> ... </body>` section. You will be adding code to this section. In particular, you will add the following:

```
<div id="slideshow">
    <div><img src="images/file1.jpg"><br><p> Caption 1.</p></div>
    <div><img src="images/file2.jpg"><br><p> Caption 2.</p></div>
    ...
    <div><img src="images/filen.jpg"><br><p> Caption N.</p></div>
</div>
```

between the `<body>...</body>` tags. For each image in the slide show, there is a line of the form

```
 <div><img src="images/file.jpg"><br><p> Caption.</p></div>
```

where `images/file.jpg` is the image file (including directory path) and `Caption` is a caption. The tags `<br>` and `<p>...</p>` mean newline and paragraph respectively.

You will get the image filename from a CSV file that for each line contains the following:

```
filename, location, caption, keyword, keyword, ...
```

The number of keywords is not fixed and can be any number greater than 0 (you may assume there is at least 1 keyword). The `filename` refers to the filename (this may include directories) of the image, `location` refers to the place where the photo was taken, `caption` is the Caption above and keywords are descriptors of the image that will be used to select a subset of photos to be displayed.

## 3 Basic Slideshow

Complete the functions listed below and make sure they are working perfectly before moving onto the next section. You will need to complete the following functions as described in the starter.py file and save the file as `slideshow.py`.

| | |
|---|---|
| read_file(file) | Given an open file, return the contents as a string. |
| get_header(str) | Given a `str`, return the entire substring starting at the beginning of the `str` up to and including `</head>`. Requirement: The string contains `</head>`. |
| add_div(str, str) | Return a string that is:<br>`<div><img src="filename"><br><p>caption</p></div>`<br>where filename is replaced by the first parameter and caption is replaced by the second parameter. |
| create_image_dict(file) | Given an open csv file with the format:<br>`filename,location,date,caption,keywords, ...`<br>Return a dictionary with key filename and value [location, date, caption] |
| add_body(dict, str, list = None) | Given an image dictionary where each key is a filename and each value is a list is of the form [location, date, caption], a string that contains the header of an html file `<html><header>...</header>`, and an optional third parameter, return a string that contains the header, the body (including the slideshow div) and the end `</html>` tag of the slideshow html file. The body should contain one ¡div¿ statement for each filename in the dictionary. If a third parameter, a list of filenames, is passed, include only those filenames in the list and display them in the same order as specified by the list. |

After completing the above functions you should write a `main` program that uses the above functions to create a webpage slideshow. You can try your own images if you like. Samples of the following files can be found on the course website.

- The input HTML file you should use is called empty_slideshow.html.
- The CSV file that contains the image files and information should be called files.csv.
- Feel free to edit the sample given to have more pictures, keywords etc.
- Sample image files are also on the website. You should download and place in a subdirectory called `images` of the directory where your assignment files are.

The demo slideshow can be loaded into a browser to see what it should look like. To see how the html code should look, open the file in Wing.

## 4 Selecting Images

We now add a couple functions that allow the user to specify which images they would like to view, for example, they may wish to see all images having to do with the keyword "sunset" or all images in chronological order. Complete these functions in `slideshow.py`. Notice that several of them were in `e6.py`. Sample solutions to the functions from `e6.py` will be posted on Monday in case you were unable to get them working.

| | |
|---|---|
| create_keyword_dict(file) | Given an open csv file with the format: filename,location,date,caption,keywords,keywords, ... return a new dictionary where the key is a keyword and each value is a list of filenames that have that keyword in their list of keywords. |
| create_date_dict(dict) | Given an image dictionary, return a new dictionary where the key is a date and the value is a list of filenames of images taken on that date. |
| sorted_images(dict) | Given an image dictionary return a list of the filenames sorted by date. |
| find_images_with_keyword(dict, list) | Given a keyword_dictionary and a list of keywords, return a list of filenames associated with the given keywords. The list of keywords should include all filenames having one or more the of the specified keywords (ie, a filename may be only associated with one of the keywords in the list). |
| intersect_lists(list, list) | Given two lists, return the first list with the items not in the second list removed from the first list. For example intersect_lists([1, 2, 3, 4, 5, 6], [4, 2, 6]) would return [2, 4, 6]. |

## 5 Main Program

Add prompts to your `main` program to ask the user for as many keywords as they like, one at a time until they simply press enter returning an empty string - keywords should not be case sensitive (ie, Dog and dog should both return the same thing). If the user only inputs an empty string then your program should include all images. Note this requires using a `while` loop. After asking the user for their keywords, your program should ask the user if they would like the slideshow to be in chronological order. Their answer should be "yes" or "no".

You should use the following questions:

```
Please enter a keyword.  Enter nothing to quit.

Slide show in chronological order?  (yes/no)
```

Your program should write the slideshow to a file `slideshow.html` containing only those images satisfying specified keywords and in chronological order if required.