Stage 1

For the first few questions, we'll ask you to list files. These questions have a few things in common:

- any files with names beginning with "." should be excluded
- you must produce a list
- the list must be in reverse-alphabetical order

Some things will vary:

- which directory you'll look at
- whether the list contains simples file names, or paths
- sometimes you'll need to filter to only show files with certain extensions

You may consider writing a single function to answer several questions (hint: things that change for different questions can often be represented with parameters).

Question 1: What are the names of the files present in the `sample_data` directory?

Hint: Look into the `os.listdir` function. Produce a list of file names, sorted in **reverse-alphabetical** order.

Question 2: What are the paths of all the files in the `sample_data` directory?

In order to achieve this, you need to use the `os.path.join()` function. Please do not hardcode "/" or "\" because doing so will cause your function to fail on a computer that's not using the same operating system as yours.

Question 3: What are the paths of all the files in the `full_data` directory?

Question 4: What are the paths of the CSV and JSON files present in the `sample_data` directory?

Filter to only include files ending in `.csv` or `.json`.

To clarify, this function must do everything you did for Question 2, as well as the additional step above.

Question 5: What are the paths of the CSV and JSON files present in the `full_data` directory?

For the following questions, you'll need to create a new Tweet type (using namedtuple). It will have the following attributes:

- tweet_id (string)

- username (string)
- num_liked (int)
- length (int)

Please ensure you define your namedtuple exactly according to the specifications above, or you will be unable to pass the tests. You should be able to use your Tweet type to create new Tweet objects, like this:

```
t = Tweet("id123", "user456", 100, 140)
t
```

Running the above in a cell should produce output like this:

```
Tweet(tweet_id='id123', username='user456', num_liked=100, length=140)
```

Question 6: What are the tweets present in the CSV file `1.csv` in `sample_data`?

Your goal for this question is to write a function to parse a CSV file, constructing a **list of Tweet objects**, where each row of the CSV file corresponds to one Tweet.

For example, here's what the first (non-header) row of the `1.csv` in `sample_data` file looks like:

```
1467811372,Mon Apr 06 22:20:00 PDT 2009,USERID_6,5882,@Kwesidei not the whole crew
,True
```

The corresponding Tweet object should look like this (note that `length` is the number of characters in `tweet_text`):

```
Tweet(tweet_id='1467811372', username='USERID_6', num_liked=5882, length=29)
```

Notice that we're ignoring a few fields from the CSV, such as `date` and `is_retweet`.

Question 7: What are the tweets present in the CSV file `2.csv` in `sample_data`?

Question 8: What are the tweets present in the CSV file `1.csv` in `full_data`?

Question 9: What are the tweets present in the CSV file `2.csv` in `full_data`?

If you just tried to run your code as-is on this file, chances are it crashed, or you had some missing data. This is because some of the rows in this file, are incomplete or inconsistent in some way. You must now go back and modify your CSV parsing function to deal with situations like this.

In short, whenever you see a row in the CSV file which does not have all the fields present, just skip that row and move on to the next one, parsing what remains.

Question 10: What are the tweets present in the JSON file `1.json` in `sample_data`?

Just like before with the CSV files, we're going to now parse a JSON file and convert it to a list of Tweets, so that all of our data from different files is going into one common format that's easy for us to work with.

The JSON files we have have the data saved as one big dictionary, with the keys in the dictionary being the tweet_id, and the values being a smaller dictionary, containing all the details of the tweet with that tweet_id. Feel free to open up a JSON file and take a look at it to get a sense of how it's structured (this is always a great first step when you're trying to parse data you're unfamiliar with).

Your task here is to convert each JSON file to a **list of Tweet objects** (similar to what we did when parsing the CSVs). Each key-value pair in our big dictionary therefore corresponds to one namedtuple in the list.

Here's the first tweet in the JSON file, `1.json` in `sample_data`

```
{
  "1467810369": {
    "date": "Mon Apr 06 22:19:45 PDT 2009",
    "username": "USERID_4",
    "tweet_text": "@switchfoot http://twitpic.com/2y1zl - Awww, that's a bummer.
You shoulda got David Carr of Third Day to do it. ;D",
    "is_retweet": false,
    "num_liked": 315
},
```

And here's the corresponding namedtuple:

`Tweet(tweet_id='1467810369', username='USERID_4', num_liked=315, length=115)`

Question 11: What are the tweets present in the JSON file `2.json` in `sample_data`?

Question 12: What are the tweets present in the JSON file `5.json` in `full_data`?

Question 13: What are the tweets present in the JSON file `1.json` in `full_data`?

Once again, we have some JSON files that are broken, such as this one. Unfortunately, unlike CSV files, broken JSON files are much more complicated to fix, so we can't just skip over one tweet and salvage the rest. Instead, your JSON parsing function should skip any file it cannot parse using `json.load` and just return an empty list.

Question 14: Which file in the directory `sample_data` contains the tweet with tweet_id '1467813137'?

Produce the **path to the file**. If you can't find this tweet_id in any of the files in this directory, produce `False`.

Hint: Use the functions you've written to help you accomplish this task, as it involves a combination of looking through all the files in a folder, parsing them, and then looking through the parsed list.

Question 15: Which file in the directory `full_data` contains the tweet with tweet_id '1467862937'?

Question 16: Which file in the directory `full_data` contains the tweet with tweet_id '1467907751'?

Question 17: Which files in the directory `sample_data` contain tweets by the user "USERID_1"?

Be sure to produce a **list of paths** (even if it's just 1 path) sorted in **reverse-alphabetical order**.

Question 18: What are the tweets present in all the files in the `sample_data` directory?

Produce a single **list of Tweets** containing all the tweets, sorted in **ascending order by tweet_id**.

Question 19: What are the tweets present in all the files in the `sample_data` directory, sorted by num_liked?

Produce a single **list of Tweets** containing all the tweets, sorted in **descending order by num_liked**.

Question 20: What are the first 20 tweets present in all the files in the `full_data` directory, sorted by num_liked?

Produce a single **list of Tweets** of length 20 containing the first 20 tweets sorted in **descending order by num_liked**.

That's it for Stage 1. In the next stage, we'll begin using the data structures we've set up to do some analysis that spans across multiple files!