CS100: Project 2

Revision Date: October 2, 2015

Preamble

> "What a wee little part of a person's life are his acts and his words! His real life is led in his head, and is known to none but himself." – Mark Twain

You may develop your code anywhere, but you must ensure it runs correctly under a Linux distribution before submission.

Twitter Feeds

Your task is to read in two files containing twitter feeds, merge them in reverse chronological order, and provide some basic summary information about the files.

The test input files can be obtained with these command:

```
wget troll.cs.ua.edu/cs100/python/projects/tweet1.txt
wget troll.cs.ua.edu/cs100/python/projects/tweet2.txt
```

You are to write a python program to accomplish this task. The names of both files will be passed to your program using the command line. You can read about how to handle command-line in the chapter entitled "Input and Output" in the textbook.

The output of your program should be two parts. First, the program should print to the terminal the name of the file that contained the most tweets followed by the number of tweets tweeted. In the event of a tie, print both filenames along with the number of tweets (Note: a file may be empty). Second, the program should also print the five earliest tweets along with the tweeter. Finally, the program should write to a file *output*, the lines from the inputted files sorted in reverse chronological order (most recent tweets first and earliest tweets at the end).

The name of your program will be *twittersort.py*.

Reading from files

Once you get the name of a file from the command line, you will need to read the text in the given file and build a record for each line in the file. To read the text in a file, you should use the *Scanner* class, which you can retrieve with:

```
wget http://troll.cs.ua.edu/cs100/python/projects/scanner.py
```

Run this command in the same directory as your project files. Read the chapter entitled "More on Input" in the textbook on how to read in a file. You should use the proper function for reading each field of the record. For instance, note that the *readtoken* function of a *Scanner* object will break up a quoted string into individual tokens. To get the full tweet without breaking up the string, you will need to use a different function.

File Format

Each input file will contain a list of records with one record appearing on each line of the file. The format of a record is as follows:

```
@TWEETER "TWEET" YEAR MONTH DAY HR:MN:SC
```

Your job will be to read in each file and for each line in the file, create a record with the above information. In the above format, a tweet is a string that can contain a list of tokens. Also, HR:MN:SC should be treated as a single field of the record, the time. Note, you should remove the '@' symbol from each tweeter's name.

An Example

Suppose the two files each contained a single line.

File 1:

```
@poptardsarefamous "Sometimes I wonder 2 == b or !(2 == b)" 2013 10 1 13:46:42
@nohw4me "i have no idea what my cs prof is saying" 2013 10 1 12:07:14
@pythondiva "My memory is great <3 64GB android" 2013 10 1 10:36:11
@enigma "im so clever, my code is even unreadable to me!" 2013 10 1 09:27:00
```

File 2:

```
@ocd_programmer "140 character limit? so i cant write my variable names" 2013 10 1 13:18:01
@caffeine4life "BBBBZZZZzzzzzZZZZZZZzzzZZzzZzzZzTTTTttt" 2011 10 2 02:53:47
```

Then, running the program:

```
$ python3 twittersort.py file1 file2
```

should produce the following output to the terminal:

```
Reading files...
file1.txt contained the most tweets with 4.
Merging files...
Writing files...
Files Written. Displaying 5 earliest tweeters and tweets.
ocd_programmer "140 character limit? so i cant write my variable names"
nohw4me "i have no idea what my cs prof is saying"
pythondiva "My memory is great <3 64GB android"
enigma "im so clever, my code is even unreadable to me!"
caffeine4life "BBBBZZZZzzzzzZZZZZZZzzzZZzzZzzZzTTTTttt"
```

and the outputted text file *output* should contain:

```
poptardsarefamous   "Sometimes I wonder 2 == b or !(2 == b)"    2013 10 1 13:46:42
ocd_programmer  "140 character limit? so i cant write my variable names"    2013 10 1 13:18:01
nohw4me "i have no idea what my cs prof is saying"  2013 10 1 12:07:14
pythondiva  "My memory is great <3 64GB android"    2013 10 1 10:36:11
enigma  "im so clever, my code is even unreadable to me!"   2013 10 1 09:27:00
caffeine4life   "BBBBZZZZzzzzzZZZZZZZzzzZZzzZzzZzTTTTttt"   2011 10 2 02:53:47
```

Program Organization

Your *twittersort.py* program should have a *main* function, of course, plus the following helper functions:

- a function that compares two records based on date and returns true if the first record is more recent than the second and false otherwise
- a function that merges two arrays of records based placing more recent records before earlier records and returns the merged records as a single array
- a function that given a filename creates a *Scanner* object and creates a record for each line in the file and returns an array containing the records
- a function that takes in a Scanner object and creates a record then returns an array representing the record; note, the '@' symbol should also be removed from the tweeter's name
- a function that takes in a table of records and writes to the file *output* each record on it's own line; for each record, there should be a tab (special character `'\t'`) between the name and tweet as well as between the tweet and year, between all other fields there should be a space

For more information on creating these functions, you can refer to the chapters ”’Loops”’ and ”’More on Input and Output”’.

Stepwise refinement

Write a version of the program that:

- obtains the file names from the command line
- adds the function that creates a record
- adds the function that reads the records into a table and reads the contents of each input file
- prints which file contains the most tweets and the number of tweets in the file
- adds the function that compares two records based on date
- adds the function that merges two tables and merges the tables containing the records from each input file
- adds the function that writes the records to an output file and writes the results of the merged tables of records

Challenge

Read about dictionaries. Learn how to use dictionaries to keep track of the number of times each has tag appears in the two input files. Hashtags are common tokens in social media that start with a '#' and are followed by a string of words (such as '#thisisahashtag'). Print the most common hashtag. In addition, you can try the same with words, find the most common words in the tweets.

In addition, try to figure out how many tweets go over the 140 character limit set by Twitter. Figure out how many tweets are "short" tweets with character ranges under 50 characters long. Keep track of all the character lengths for every tweet and at the end report the average character length for a tweet.

Compliance Instructions

You should be able to run your program, like this:

```
python3 twittersort.py <file1> <file2>
```

Where `<file1>` and `<file2>` refer to the names of your input files.

Submission Instructions

Make sure you have the following files in your directory before submitting:

- twittersort.py
- scanner.py
- tweet1.txt
- tweet2.txt

Change to the directory containing your assignment. To submit the project, run the command:

```
submit cs100 YYY project2
```

Replace *YYY* with the name of your instructor. If your instructor teaches more than one section, add the hour of your class after your instructor's name.