CSE 231 **Spring 2014**

Computer Project #8

Assignment Overview

This assignment focuses on the design, implementation and testing of a Python program to process collections of data using data structures, as described below.

It is worth 60 points (6% of course grade) and must be completed no later than 11:59 PM on Monday, March 31.

Assignment Deliverable

The deliverable for this assignment is the following file:

`proj08.py` – your source code program

Be sure to use the specified file name and to submit it for grading via the **handin system** before the project deadline.

Assignment Background

Whenever a Wikipedia article is edited, information about the revision is logged by the system. The SNAP project at Stanford University has collected some of those revisions and published them in an accessible format.

The SNAP data file contains one log entry for each revision, where each log entry contains 13 lines. A sample log entry is shown below:

```
REVISION 4781981 72390319 Steven_Strogatz 2006-08-28T14:11:16Z SmackBot 433328
CATEGORY American_mathematicians
IMAGE
MAIN Boston_University MIT Harvard_University Cornell_University
TALK
USER
USER_TALK
OTHER De:Steven_Strogatz Es:Steven_Strogatz
EXTERNAL http://www.edge.org/3rd_culture/bios/strogatz.html
TEMPLATE Cite_book Cite_book Cite_journal
COMMENT ISBN formatting &/or general fixes using [[WP:AWB|AWB]]
MINOR 1
TEXTDATA 229
```

Each line of a log entry begins with a label, such as “REVISION” or “CATEGORY”. There is a blank line between log entries.

The “REVISION” line of a log entry contains the following information:

Article identification number
Revision identification number
Article title
Revision time stamp
User name of editor
User identification number of editor

The last two fields contain an Internet Protocol address (IP address) if the editor is not a registered Wikipedia user.

Assignment Specifications

1. You will develop a Python program to manage information about the revision history for Wikipedia articles. The program will input a collection of log entries from a file, and then allow the user to display statistics about the revisions.

2. The program will recognize the following commands:

QUIT
HELP
INPUT filename
TOP n EDITORS
TOP n EDITS
TOP n ARTICLES

The program will be operated interactively: it will prompt the user and accept commands from the keyboard. The program will recognize commands entered with any mix of upper and lower case letters.

If the user enters an invalid command, the program will display an appropriate message and prompt the user to enter another command.

3. The “QUIT” command will halt execution.

4. The "HELP" command will display information to the user about the commands recognized by the program.

5. The "INPUT" command will be followed by a string representing the name of an input file. The program will discard the current data set stored in memory, and then process the input file as the source for a new data set.

If the user enters an invalid file name, the program will display an appropriate message and prompt the user to enter another command.

The program will ignore all lines except properly formatted “REVISION” lines.

The program will use the data set to construct two dictionaries, as described below.

a) Each entry in the first dictionary has a key which is an editor and a value which is a set (or list) of articles revised by that editor:

```
{editor: {set of articles revised by editor}}
```

The rest of your program will be easier if that set (or list) contains no duplicates.

b) Each entry in the second dictionary has a key which is an article and a value which is a tuple (or list) that contains the number of edits and a set (or list) of editors who revised the article:

```
{article: (count of edits, {set of editors who revised the article})}
```

The rest of your program will be easier if that set (or list) of editors contains no duplicates.

6. The "TOP" command will be followed by an integer number and a string (one of "EDITORS", "EDITS" or "ARTICLES").

If the user enters an invalid command, the program will display an appropriate message and prompt the user to enter another command; the program will not display an empty report.

For each report, the program will display the relevant information about the top "n" items in a given category:

EDITORS – the editors who have revised the most articles (display the user name of the editor and the number of articles revised by that editor).

EDITS – the articles which have been revised the most often (display the title of the article and the number of times that article was revised).

ARTICLES – the articles which have been revised by the most editors (display the title of the article and the number of editors who have revised that article).

The information will be displayed in tabular form. The fields will be identified using column headers, and the fields will be aligned beneath the headers.

The reports will be sorted from highest to lowest value in the category. Entries with the same “sort” value will be displayed in alphabetical order.

7. The program will display appropriate messages to inform the user about any unusual circumstances.

8. The program will consist of at least four meaningful functions:

A function which processes “INPUT filename” commands (reads the contents of the file and builds the two dictionaries).

A function which processes “TOP n EDITORS” commands (displays the information).

A function which processes “TOP n EDITS” commands (displays the information).

A function which processes “TOP n ARTICLES” commands (displays the information).

Communication between functions will occur via parameters and return values; you may not use any global variables.

Assignment Notes

1. The project directory contains two sample data files (“sample1.txt” and “sample2.txt”) which may be used as you develop your solution. However, your program is expected to execute correctly for any properly formatted data file.

2. Additional information about the SNAP project can be found on the web:

http://snap.stanford.edu/

3. You may assume that the Article identification number and the Article title both uniquely identify an article. Similarly, you may assume that the User name and User identification number uniquely identify an editor.

4. As noted above, some “REVISION” lines contain IP addresses, rather than User names and User identification numbers (for example, `ip:24.188.31.147).` Your program will accept them as user names since they are reasonable aliases for actual people (that is not precise, but sufficient for our purposes).

5. The only time program execution will halt is when the user enters the “QUIT” command.

6. As noted above, your program must consist of at least four meaningful functions; you certainly may develop more than four functions. You will have to decide how to decompose the overall program into smaller tasks. Note that it would be natural to implement some (or even all) of the subtasks as functions.

Each function must be declared at the global level. Each function should have a coherent purpose which can be expressed succinctly in a line or two.

You may not use any global variables in your program. All communication between the functions which constitute your program will be done using parameters and return values.

Reminders

- Solve the problem using pencil and paper first.

- Never show your program source code to another student or discuss specifics of how you implemented parts of your solution at the level of Python source code.

- Develop a simple version of the program, then add functionality incrementally, testing your program thoroughly after each addition.

- Use the **handin system** to turn in each version of your program, including the final version.

- You would be wise to back up your file on your H: drive, in addition to submitting it via the **handin system**.

- Be sure to log out when you leave the room, if you’re working in a public lab.