60-141 Winter 2018

BONUS - Short Assignment 4: Dynamic Linked Lists

Due: Friday April 6, 2018 11:59pm EST

Description

Similar to the previous assignment, we will reuse the same structure with minor modifications.

Recall the simple twitter app from the previous assignment.
A "tweet" is a special object that is the fundamental building block of Twitter.
If you are curious about the composition of a tweet you can familiarize yourself with the developer pages available here:
https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/tweet-object

You can see that the tweet is quite a complex object, which is why we would focus instead on a much simpler version called "Simple Tweet". It is a good start for us to create a simple twitter application.

A simple tweet will be composed of the following:

id	int; which uniquely identifies the simple tweet
created_at	char[51]; that would store the UTC time and date.

Note that UTC is the coordinated universal time, which is a common standard on the Internet.

text	char[141]; allow up to 140 characters.
user	char[21]; the user name of the person who posted the tweet

The requirement of this assignment is to:

1. Create a ***self-referential*** structure definition called **tweet** in C that contains the simple tweet members as described above. Document it properly (Document the structure definition stating its purpose, and each member attribute). [1pts; improper documentation minus up to 2pts]
2. Create an alias using **typedef** in C for a tweet call it **<u>T</u>weet**. [1pts]
3. In the file (global) scope, create a **single dynamic event list pointers** to maintain a dynamic list of tweets. Call your pointers **ptrFirst** and **ptrLast**. Initially both pointers are NULL indicating it is an empty list [1pts]

Main functionality:
Write a program that can read the sequential text file **tweets.txt**, sorts its contents by user in alphabetical order, then write it back, overwriting the file contents with the newly sorted list.
Note that your program should work for a file of any size; so do not use an array like you did in assignment 3, use a dynamic linked list instead!
[Marks: Load the file=2pts, Sort (could use insertion sort and sort it as you load it)=2pts, save the file=2pts; clean-up memory=1pts]

Sample Input File: tweets.txt (created using assignment 3)

```
1000
Fri Mar  9 22:48:25 2018
Jim Smith
my first tweet
```

```
1001
Fri Mar  9 22:48:32 2018
Angela Smith
my second tweet
```

Sample Output File tweets.txt after sorting:

```
1001
Fri Mar  9 22:48:32 2018
Angela Smith
my second tweet
1000
Fri Mar  9 22:48:25 2018
Jim Smith
my first tweet
```

Create the following script:

```
cp assign4.c assign4.bak      (back up your C code first!)
script assign4.txt
cat assign4.c
cc assign4.c
cat tweets.txt          (this will display the original file – unsorted)
./a.out
cat tweets.txt          (this will display the new file – sorted)
exit
```

`Submit your C code and the script file (assign4.c and assign4.txt)`