CIS 550 Homework 2: Trendtracker

Due Sunday, February 11 at 11:59 PM

1 Introduction

The Twitter website has become a de facto first source for many important events in the last decade. Twitter's hashtag feature lets users tag tweets with single words or phrases (e.g. #superbowl, #algorithms, or #vacaciones). Popular or *trending* hashtags indicate strong shared interest by many people in a topic, and tracking these trends is of interest to businesses, news outlets, and researchers.

In this homework, you'll implement an array-based data structure that tracks information about a collection of hashtags, including which are most popular, i.e. are *trending*.[1] Note hashtags in the ArrayList are required to be sorted.

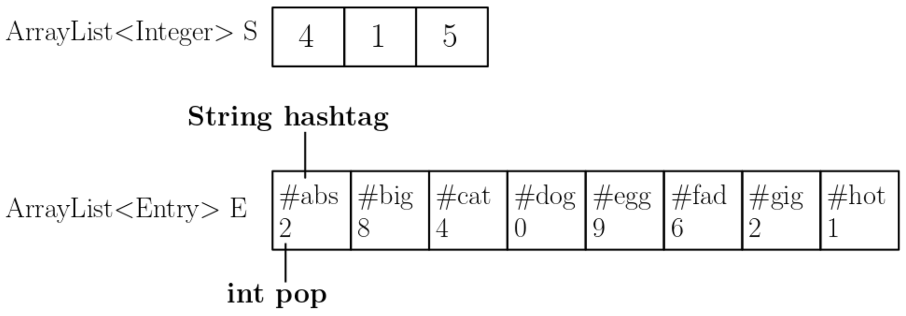

Figure 1: Representing hashtags and their popularities using an ArrayList E; S stores indices of the (up to) three most-tweeted entries in E in order from most-tweeted to least-tweeted.

2 Instructions

The following files have been given to you:

1. A class file (`trendtracker.java`) declaring the `trendtracker` class.
2. A source file (`MainTest.java`) containing a `main` function with tests.
3. A text file (`tiny.txt`) containing 1 hashtag.
4. A text file (`small.txt`) containing 4 hashtags.
5. A text file (`hashtags.txt`) containing 300000 hashtags.[2]
6. A text file (`tweeted.txt`) containing 1500000 hashtags.[3]

Download the files on the blackboard. Implements the `trendtracker` class by filling every function as required, so that `trendtracker.java` and the provided files compile into a program that runs with **no failed tests**. Submit only source file `trendtracker.java`.

[1]The trending topics displayed on Twitter's website are actually chosen using more complex algorithm.
[2]Source: `http://norvig.com/ngrams/count_1w.txt`.
[3]Source: `http://norvig.com/ngrams/count_1w.txt`.

3 Submission and Grading

Submit the aforementioned source file(s) via Blackboard as attached file(s). In the case of multiple submissions, the last submission before the deadline is graded.

For grading, each submission is compiled with the provided files and run. Submissions that do not run to completion (i.e., fail to print "Assignment complete.") receive no credit. Submissions that take an unreasonable amount of time (e.g. more than a minute or so) to run and do not meet the asymptotic efficiency requirements receive no credit. Failed tests indicate that your functions have computation errors. In this case, you will get deduction. All other submissions receive full credit.

See the course late work policy for information about receiving partial credit for late submissions.