CS2040 Data Structures and Algorithms: Problem Set 2

Due: Please check Coursemology

Harold Soh

This assignment is GRADED. Please take note of the due date and submit your solutions on Coursemology.

Overview Nowadays, it is common to work with "Big Data". In this homework assignment, we have a large database to process. The goal is to extract some useful information out of the database without having to read the entire big database into memory. The fewer the number of data accesses, the better.

Collaboration Policy. You are encouraged to work with other students on solving these problems. However, you **must** write up your solution **by yourself**. We may, randomly, ask you questions about your solution, and if you cannot answer them, we will assume you have cheated. In addition, when you write up your solution, you **must** list the names of every collaborator, that is, every other person that you talked to about the problem (even if you only discussed it briefly). Any deviation from this policy will be considered cheating, and will be punished severely, including referral to the NUS Board of Discipline.

Herbert the Robotic Clown

(Total: 30 points) As we all know, the Acme Corporation is a world leader in making people smile. (Motto: *For fifty years, the leader in creative mayhem!*) The Acme Corporation is well-known for its wide selection of anvils and birdseed, along with specialty items such as dehydrated boulders and earthquake pills. However, in recent days, Acme stock has been depressed, and their inventors have been hard at work coming up with new ways to make people happy.

Their most recent invention is a robotic clown named *Herbert*. He is a fully automated clown, well equipped with all of your favorite Acme gizmos, and he can be hired to perform at your party! While Herbert is an excellent clown, he is not so good at math. (He was programmed with a very funny joke about how $1+1=3$, and ever since he has had great difficulties.) Your job, in this problem, is to help Herbert with the financial aspects of his job.

Problem 1.1 (20 points) Herbert is paid by the minute. Sometimes, he is hired for a very short performance (perhaps, only 1 minute long), and sometimes he is hired for a long engagement that may last for weeks. Herbert's salary, however, is very complicated: he is paid different amounts for different lengths of time. (An additional complication is that Herbert is payed in SmileBucks, which Herbert then may exchange for spare parts and electricity. Currency conversion and bartering techniques are outside the scope of this problem set.)

Herbert has given you an employment log of his to analyze. The log is very long, containing data from the last several months. Each line of the log represents one minute of work. Hence a job that takes ten minutes is represented by ten lines in the log.

Each line in the log consists of two components: the name of the employer, and the total amount of SmileBucks (SB) that Herbert has been paid for the job, up to and including that minute. For example, consider the following two jobs where Humperdink hires Herbert for 5 minutes and Hortensia hires Herbert for 2 minutes:

```
Humperdink:7
Humperdink:10
Humperdink:12
Humperdink:14
Humperdink:16
Hortensia:10
Hortensia:11
```

Here, we see that Herbert was paid 7 SB by Humperdink for the first minute. He was paid an additional 3 SB for the second minute, yielding a total of 10 SB. In total, for the five minutes, he was paid 16 SB. Herbert then was paid 11 SB by Hortensia. You can assume that each employer only hires Herbert once, i.e., all the names of the employers are unique.

You have been provided with a file `HerbertLog.java` which contains the `HerbertLog` class. This class has a constructor, which takes one argument: the filename containing the log. It provides you with two methods:

- `long numMinutes()`: returns the total number of minutes in the log file.
- `Record get(long i)`: return the record associated with minute i.

Notice that the record returned by the `get` method is an object of type `Record` (see the provided `Record.java` class).

Herbert the Robotic Clown continued on next page...

Your job is to add a new method to the `Clownculator` class (in `Clownculator.java`) that calculates the total amount of money made by Herbert over the duration of that log. In other words, create a new method `long calculateSalary()` that returns the total salary of Herbert.

Notice that a simple solution involves reading every single line of the log. However, the log might be very long (even if it contains relatively few jobs). Devise an algorithm which retrieves as few records as possible. (That is, call the `get(long i)` method the minimum number of times.) Partial credit will be given for any working solution, even if inefficient. More credit will be given for better (more efficient) solutions.

In order to test your solutions, we have released various logs containing Herbert's history. Report the number of `get` operations you require for each of the sample files. Use the `ClownculatorTest.java` test to check your answers.

Problem 1.2 (10 points) Herbert has one flaw: he is lazy. He does not want to work so many long, long jobs. He would prefer that none of his jobs be more than ten minutes. Or perhaps, 20 minutes? What should be the maximum length job that Herbert performs? He needs to make at least enough SmileBucks every month to pay for his spare parts!

Your job is to create a new method: `long calculateMinimumWork(long goalIncome).` This method returns the *minimum* number of minutes m such that if Herbert works no more than m minutes for each employer, then he makes at least the specified incoming. Consider the following example:

```
Amelia:10
Amelia:20
Amelia:30
Bert:40
Chloe:1
Chloe:2
Chloe:3
```

If the desired income is 60, then the method should return 2: if Herbert works for at most 2 minutes for each employer, then he will make 62 SB. If he only worked 1 minute for each employer, then he would make only 51 SB, which is not enough. (Notice that the most he can make in this case, by working for up to 3 minutes per employer, is 73.). If the desired income cannot be attained, then the method should return -1.

Again, minimize the number of times that you query the log. Partial credit will be given for slow solutions (in this case, as long as they do not read every line in the log), but faster solutions that read the log fewer times will get more credit.

In order to test your solutions, we have released various logs containing Herbert's history. Report the number of `get` operations you require for each of the sample files. Use the test file `ClownculatorTest.java` to check your answers.

— END OF PROBLEM SET 2 —