CSE 2010, HW4, Fall 2022

Due Thu Oct 20 at the start of your lab section; Submit Server: class = cse2010, assignment = hw4S*x*Individual

Due Thu Oct 20 at the end of your lab section; Submit Server: class = cse2010, assignment = hw4S*x*GroupHelp

x is 12 or 34—your section number (or "c" for c).

To improve the experience of customers and potentially revenue, an online music store would like to recommend music that the customers might like. Different customers have different taste. How would you customize the music recommendation for a customer?

The goal of HW4 is to recommend music for a customer based on another customer with the most similar taste. Since the store collects ratings for songs from each customer, it could find the customer with the closest ratings and recommend songs that the target customer has not rated. Each rating is between 1 (poor) and 5 (excellent). The overall algorithm has two main tasks:

1. For a target customer *targetCust*, find the customer *closestCust* with the closest (discussed later) song ratings. If more than one customer are equally close, choose the customer who is alphabetically earlier.
2. Recommend song(s) that *closestCust* gave the highest rating and he/she likes the song(s) (ie. rating ≥ 4), but *targetCust* has not rated. For example, *closestCust* rated *song*0 and *song*1 with 5 and *song*2 with 4, and *targetCust* did not rate the 3 songs, *song*0, *song*1, and *song*2 are recommended. Also, if *closestCust* did not rate any songs or did not like (rating ≤ 3) any songs that *targetCust* did not rate, *closestCust* is updated to the next closest customer.

Let S be the set of songs that both customers x and y have rated and $rating_x(s)$ be the rating of song s from customer x. If S is empty, distance cannot be calculated. Otherwise, we can calculate the distance between x and y based on two factors: (1) the average absolute difference in ratings: $\frac{1}{|S|}\sum_{s\in S}|rating_x(s) - rating_y(s)|$, and (2) the reciprocal of the number of songs they both rated: $\frac{1}{|S|}$. Combining the two factors:

$$distance(x,y) = \frac{1}{|S|} + \frac{1}{|S|}\sum_{s\in S}|rating_x(s) - rating_y(s)|. \quad (1)$$

To store the customer names and ratings, use an array of pointers to `Customer` (java objects) that includes a name and ratings of songs. For simplicity, assume at most 100 customers, at most 10 letters in a customer name, and only 10 songs (numbered 0-9). Functions/methods include:

- insertCustomer(name, ratings) // maintain alphabetical order
- findCustomer(name) // use binary search

To manage and find the closest customer efficiently, use a priority queue implemented with a heap. Each entry/item has: key (distance) and value (customer name). Assume at most 100 entries. A tie in the key is broken by the value (alphabetically). Functions/methods include:

- insert(entry)
- removeMin()
- getMin()
- isFull()
- isEmpty()

To implement the priority queue, you may modify/rewrite Code Fragment 9.8 on p. 377-378. We will evaluate your submissions on code01.fit.edu so we strongly recommend you to test your programs on code01.fit.edu. To preserve invisible characters, we strongly recommend you to download and save, NOT copy and paste, input data files.

Input: Input is from the command-line arguments for HW4.java in this order:

1. Filename of initial song ratings. The first line has the target customer name. Each of the follwing lines has the name of a customer and 10 song ratings. Each rating is between 1 (poor) and 5 (excellent), or 0 (not rated).
2. Filename of actions, each line has one of the following actions:
   - AddCustomer *custName* *rating*0 *rating*1 ... *rating*9
   - RecommendSongs
   - PrintCustomerDistanceRatings

Output: Output goes to the standard output (screen), each line corresponds to an action:

- AddCustomer *custName* *rating*0 *rating*1 ... *rating*9
- RecommendSongs *closestCustName/none* *song*1 *rating*1 *song*2 *rating*2 ...
- PrintCustomerDistanceRatings

```
12345678901234567890123456789012345678 //show spacing, not output
      Mickey     1 2 3 4 5 0 0 0 0 0
0.600 Alice      1 2 3 4 3 1 2 3 4 5
...
```

PrintCustomerRatings prints the target customer first, then the other customers in alphabetical order in a table. Distance starts at column 1, customer starts at column 7, *rating*0 is at column 18, *rating*1 is at column 20, ... Print distance with three decimal places (x.xxx), for example, 1.234 or ----- if the distance cannot be calculated.

Sample input files and output are on the course website.

Extra Credit (10 more points) For the priority queue, use bottom-up heap construction (heapifying a complete binary tree) to efficiently initialize the priority queue from data in the first input file. Add function/method initialize(entryArray, numEntries) near the top of one of your program files.

Submission: Submit HW4.java (and HW4Extra.java for extra credit) that has the main method and other program files. Submissions for Individual and GroupHelp have the same guidelines as HW1. Note the late penalty on the syllabus if you submit after the due date and time as specified at the top of the assignment. Late submission is NOT available for extra credit.