CSE 2010, HW1, Spring 2019

Due Thu Jan 24 at the start of your lab section; Submit Server: class = cse2010, assignment = hw1S*x*Individual

Due Thu Jan 24 at the end of your lab section; Submit Server: class = cse2010, assignment = hw1S*x*GroupHelp

x is 14, 23, or j—your section number (sections 1 and 4 are combined into section 14, ..., j for java).

An online retailer has a warehouse, where workers pick items from storage shelves to fulfill customer orders. The picked items can later be packed and shipped to the customers. Each order could have a different number of items and different categories of items. To save time, a worker can handle more than one order. How would you design a system to assign workers to customer orders?

For this assignment, for simplicity:

- Assume 2 categories of items (books and electronics), 5 initially available workers in this order: Alice, Bob, Carol, David, Emily, and each order has at most 10 items.
- A worker can be assigned to more than one customer order ("bundling" the orders) if all items in the orders are in the same category, but the total number of items does not exceed 10. For example, if the items are all books and no more than 10 books in multiple orders, they may be bundled. The system should not read in the entire input file before processing orders.
- If a worker has capacity to bundle, he/she waits to potentially bundle orders, but should not wait for more than 5 minutes.
- Only consecutive orders can be bundled; ie. older orders before newer ones.
- Assume each worker needs 1 minute to find and pick each item in the same category and an additional 5 minutes of traveling time for each category.

To separately manage the **customer orders**, **available workers**, and **worker assignments**, use 3 singly linked lists. We will evaluate your submission on code01.fit.edu, so you are strongly recommended to ensure your program functions properly on code01.fit.edu.

Input: To simulate the customer orders, an input file contains the customer orders in the same directory as your program file called hw1.c (HW1.java) that has the main method. Your submission takes the input file name as a command-line argument. Each line is one of the following:

- CustomerOrder $orderTime\ customer\ numberOfBooks\ numberOfElectronics$
- PrintAvailableWorkerList $printTime$
- PrintWorkerAssignmentList $printTime$
- PrintMaxFulfillmentTime $printTime$

For simplicity, time is in HHMM format (HH: 00-23 and MM: 00-59), the leading zero is optional. Fulfillment Time is the amount of time between when an order is made and when a worker finishes picking items for the order(s). MaxFulfillmentTime is the longest Fulfillment Time among all the orders so far. AvailableWorkerList is in availability order. Ordering of WorkerAssignmentList is the expected completion time of a worker.

Output: The program prints events to the standard output (screen). Each event is on one line and possible events are:

- CustomerOrder $orderTime\ customer\ numOfBooks\ numOfElectronics$
- WorkerAssignment $assignmentTime\ worker\ customer1\ customer2$ ...
- OrderCompletion $orderCompletionTime\ customer$
- AvailableWorkerList $worker1\ worker2$ ...
- WorkerAssignmentList $worker1$:$customer1\ worker2$:$customer2a$,$customer2b$ ...
- MaxFullfillmentTime $numberOfMinutes$

Submission: Submit hw1.c (HW1.java) that has the main method, (modified or your own) SinglyLinkedList.java and other program files. Submissions from individual students are due at the beginning of their respective lab sections via assignment *hw1SxIndividual* (see the top).

During the lab session on the due date, we encourage students to bring test cases (beyond the sample input) to test and improve each other's program in the group. Improved programs are submitted via assignment *hw1SxGroupHelp*, which is due at the end of the lab section (see the top). Your program is mainly evaluated based on *hw1SxIndividual*. Improvement on test cases will receive half credit. Specifically, $testCaseImprovement(hw1) = testCaseScore(hw1SxGroupHelp) - testCaseScore(hw1SxIndividual)$; $testCaseScore(hw1) = testCaseScore(hw1SxIndividual) + testCaseImprovement(hw1)/2$.

Note the late penalty on the syllabus if you submit after the due date and time as specified at the top.