CSCI-1200 Data Structures — Spring 2023
Homework 4 — Tool Rental Lists

In this assignment you will write a program to manage the inventory and customers of the Acme Tool Rental Company. Your program will handle several different operations: adding items to the inventory, renting tools to customers, and handling returned items. *Please carefully read the entire assignment before beginning your implementation.*

The input for the program will come from two files, an inventory file and a customer file, and the output will go to separate files for inventory and customers. These file names are specified by command-line arguments. Here's an example of how your program will be called:

```
rental.exe inventory_file customer_file inventory_output_file customer_output_file
```

The form of each input file is simple. Each line of the inventory file begins with an inventory ID which begins with a T and is followed by a 4 digit integer. Next is an integer quantity, followed by an item description. You may assume the input file strictly follows this format, i.e., you don't need to worry about format error-checking. However, you should check for invalid IDs. If a line that does not begin with a T or does not have a quantity greater than 0, print a message to `std::cerr` stating "`Invalid inventory ID XXXX found in the inventory file.`" where XXXX is the invalid ID.

```
T1001 1 chainsaw
T3056 6 Floor_Jack
T2001 2 Roto_Hammer_1/2_inch
```

Each line of the customer file begins with a customer ID, a C followed by a 4 digit integer. Next is an action, either the word rent or return, followed by a timestamp, a quantity, an inventory part number, and finally the customer name. The time stamp is a four digit integer. You should check for invalid IDs, for example a line that does not begin with a C. If you encounter an invalid ID, print a message to `std::cerr` stating "`Invalid customer information found for ID XXXX in the customer file.`" where XXXX is the invalid ID.

```
C0029 rent 0935 1 T1001 Emma_Watson
C0010 rent 1000 1 T2001 Yogi_Berra
C0010 rent 1000 2 T3056 Yogi_Berra
```

In your program, you will create two classes, one for customers and one for inventory items. You will also maintain two STL `lists`; an inventory list and a customers list. Do not use maps or any other structure that hasn't been covered in lecture. The inventory list should contain the items and the available quantity of that item. As customers rent or return the item, the quantity should be adjusted.

When a customer attempts to rent an item, check that sufficient quantity of the item is available. If there is a sufficient quantity, rent the requested quantity of items to the customer and adjust the inventory quantity. If the customer's request cannot be filled, do not rent any item to the customer and add the customer's request to a wait list along with the timestamp. If a customer requests an item whose inventory ID is not in the inventory, print an error message to `std::cerr` stating "`Customer CXXXX requested item TYYYY which is not in the inventory.`" CXXX is the customer ID and TYYYY is the inventory ID. You will need to keep track of what items each customer rents and any pending items for the customer. You will also need to keep track of which customers rent an item.

When an item is returned, adjust the available inventory quantity. Check the wait list for any customers requesting the item. Attempt to fill the requests in timestamp order. If a customer on the wait list has requested a quantity greater than the available amount, go on to the next customer. If the customer returning the item has no other rental items or pending items, remove the customer from the customer list using the STL `list`'s `erase` function. The customer list should only contain active customers. If a customer attempts to return an item that is not in the inventory, print an error message to `std::cerr` stating "`Customer CXXXX attempted to return item TYYYY which is not in the inventory.`" CXXX is the customer ID and TYYYY is the inventory ID. If the customer returns an item that this same customer is waiting for, adjust the inventory and customer's pending quantity. If the customer has no rented or pending items, remove the customer from the customer list. If the customer tries to return an item she has not rented, print an error message to `std::cerr` stating "`Customer CXXXX attempted to return item TYYYY which she/he did not rent.`"

The inventory list must be maintained in order by inventory ID. Similarly, the customer list must be maintained in order by customer ID. You may not use the STL `sort` functions for this assignment. Use the STL `list insert` and `erase` functions to maintain all lists in the proper order as you process each rental or return event.

After reading the input files, produce a report of the current inventory and active customers. For each inventory item, print the ID, the available quantity and the description as shown in the example below. On the next line, if any customers are renting this item, print "Rental Customers" and then print the customer IDs and names of customers renting this item, followed by the quantity that they have rented in parenthesis as shown below. If no one is currently renting the item, print nothing. On the next line, print similar information for any customers waiting for this item including the quantity they have requested in parenthesis. Here is an example of the format to follow:

```
T1001 0 available chainsaw
Rental Customers: C0009 Snoop_Dawg (1)
Pending Customers: C0001 Jane_Doe (1)

T2001 2 available Roto_Hammer_1/2_inch
Rental Customers: C0008 John_Snow (1) C0010 Yogi_Berra (1)

T2002 2 available Roto_Hammer_1_inch
```

The customer report is similar. It should contain a line for each active customer. An active customer is one with rented or pending items. The line should begin with customer ID, name, followed by lines containing rented items and quantities followed by lines for pending items and their quantities. Here is an example:

```
C0001 Jane_Doe
Pending: T1001 (1)

C0008 John_Snow
Rentals: T2001 (1)

C0009 Snoop_Dawg
Rentals: T1001 (1)
```

Sample input and output files are posted on the course web site. Please follow these examples exactly to aid in the automatic grading of your work. You can use the UNIX `diff` command to compare your output to the sample output files.

Order Notation

You should implement the functionality above with efficiency in mind. In your README.txt file, use order notation to analyze the computation needed for each stage of this program: loading the inventory, a single rental event, a single return (with or without pending customers), and the overall cost of the entire program. In your analysis, use these variables:

i = # of different inventory items
c = # of different customers
p = # of pending customers
q = max quantity of a particular tool owned by the store
r = max # of tools rented or requested by a customer at one time
p = total # of pending tool rentals
e = total # of rental or return events

Briefly justify your answers.

Additional Requirements, Hints and Suggestions

You may not use vectors, arrays, or the sort function for this assignment. Use the standard library (STL) lists and iterators instead. You may not use maps, or sets, or things we haven't discussed in lecture yet. You must write at least two new classes: one for inventory and one for customers. You may create additional helper classes.

Submission

Use good coding style when you design and implement your program. Be sure to make up new test cases and don't forget to comment your code! Please use the provided template `README.txt` file for any notes you want the grader to read. **You must do this assignment on your own, as described in the "Academic Integrity for Homework" handout. If you did discuss the problem or error messages, etc. with anyone, please list their names in your `README.txt` file.**

Extra Credit To encourage your fluency with the traditional step-by-step debugger we will post an additional Crash Course in C++ module: "Lesson 12: Debugger Use". Completing this module before Friday March 3rd at 11:59pm will be worth a *small* number of extra credit points on Homework 4.