CS256 Lab 9 - Get in line

Value: 30 points, plus 10 points extra credit
Due: Friday 8 November @ 23:50 (design document due for class on Wednesday 6 November)
10% bonus for submitting it before Friday @ 18:00!
Submission Format: Design document (image) and a Python script, lab_09.py, optionally lab_09_ec.py.

This lab uses a priority queue data structure to keep a bunch of things which each have a relative priority in order and then retrieve them in order. This will give us more practice designing objects, writing code, and exploring ordering, a common task in computing. In doing so we will learn more about how data structures and algorithms are intertwined and begin to explore sorting.

Setup

You are the manager of a T00ty Fruity franchise, which has a loyalty program. The more you purchase the higher number of priority you are given for the next time you visit. When customer shows-up they present their priority and are placed in-line accordingly, if there is already someone or someones in-line with the same priority they are placed at the back of that priority “section”. If they don’t have a priority they are placed at the end of the line (that is the end of the no priority section), these are indicated by a 0 priority. After a bunch of customers arrive, conveniently packaged in one data file, and are queued, you will display the first customer in line and the last one.

Tasks

Design, write and test:

- A class for an Item that holds a customer name (string) and a priority (integer). Higher numbers are higher priority, 0 is the lowest (new customer, or forgot their card, or a privacy hawk).
- A class for a Priority Queue that holds Items in order by priority, and within priority by arrival time.
- A mechanism to read a datafile of (name, priority) tuples and populate a Priority Queue with Items.
- A mechanism to display the first person in the Priority Queue.
- A mechanism to display the last person in Priority Queue.
- A mechanism to remove the first person from the Priority Queue.
- Your main() function should read the input file name from the command line, check to see that it exists, and call your populate method. After populating the Priority Queue you should
    - print the first person in the queue
    - remove the first person from the queue
    - display the new first person in the queue
    - display the last person in the queue
- Your solution should print the results to stdout in this format (these are actual values):

```
alice
carol
bob
```

Implementation Notes

- Design your solution on paper first, again there is more of your design in this lab than in previous ones. Make sure your design covers all of the assignment, you will be turning-in an image of it on Wednesday and Friday.
- Your script should have one command line argument, -i for the input file
- Your script should use the main() technique, it helps organize the code and it makes reuse much easier.

Extra Credit

- (10 points) Simulations are a very powerful tool across the Natural and Social Sciences, let's make one. The input file will now include information about arrivals and servicing, like so:
  arrive, bob, 5
  arrive, alice, 11
  service
  service
  arrive, carol, 3
  …
- Service removes the first person (i.e. the one with the highest priority) from the line.
- Your solution will need to work dynamically, as each line of the file is read and processed the contents of your Priority Queue will evolve.
- After the input file has been processed display the first and last person from the queue in this format (these are actual values):
  ```
  alice
  bob
  ```

- There will be a separate input data file for the extra credit.
- Make a copy of your base solution and riff on that, there is no risk with this route so you might as well give it a try.