Epoch - Sydney
2021 Jun

Implement a state machine for an exchange.

Exchange State Problem

```
Thank you for agreeing to submit a solution to the Epoch homework assignment.
Think of the homework as a way to demonstrate your style of design and
programming. Do as much as you have time for but we know life can be busy, don't
feel you have to spend more than a couple of hours if you don't have it. You
won't be disadvantaged. If you do want to do more, go for it! However you
approach it, we may want to talk about the solution in subsequent interviews.

The task is to write a C++ program that that tracks the trading state of various
stocks (also known as instruments) that are traded at a new Stock Exchange. The
instruments should go through the following phases each day:

PreOpen_Auction  : 08:30 - 09:00
Open             : 09:00 - 12:00

Intraday_Auction : 12:00 - 12:30
Intraday_Close   : 12:30 - 13:30
Intraday_Auction : 13:30 - 14:00

Open             : 14:00 - 15:30
Closing_Auction  : 15:30 - 16:00
Closed           : 16:00 - 08:30(next day)

The timings are not completely fixed. When a stock(instrument) in question
enters a particular phase, a notification(event) is sent from the exchange.
An instrument's trading state should not change before or after its designated
time interval. Unfortunately, it is a new exchange that has just launched. Due
to software bugs, it sometimes sends out wrong states that your program must
filter out.
Also note that this exchange may list at least two thousand instruments.

Input:
Your program should accept a filename from the command line. The file is
for one trading day and would list one exchange notification on each line. Not
all notifications might be relevant for tracking an instrument's state.
Format:
<timestamp>,<event type>,<instrument>,<event data...>

Definitions:
<timestamp> : milli seconds since midnight; will be a monotonically
              increasing unsigned integer value.
<event type>: an unsigned integer. Event type for exchange state notification
              is 2.
<instrument>: ASCII text identifier.
<event data>... : variable length and separated by commas.

The line will contain no spaces in any field.
```

```
Processing: You program should read the exchange events file and for each line:
update an instrument's current state as long as it's an allowed transition as
per the phase table above. Print a line in either case, i.e., new state after a
successful state change, or a warning if it could not.
The format of the output should be:
<timestamp>,<event type>,<instrument>,<old state>,<new state>,<Success/Failure>

Example:
-------------
28800000,2,ABC,PreOpen_Auction
28800001,0,ABC,B,100,50
28800001,2,XYZ,PreOpen_Auction
28800002,0,XYZ,S,10,508
32400000,2,ABC,Open
...
39600000,2,ABC,Open
...
43200000,2,ABC,Intraday_Auction
43200001,0,ABC,B,10,52
...
57600005,2,ABC,Closed
57600010,2,XYZ,Closed
57600012,2,XYZ,PreOpen_Auction

Output:
-------------
28800000,2,ABC,Closed,PreOpen_Auction,Success
28800001,2,XYZ,Closed,PreOpen_Auction,Success
32400000,2,ABC,PreOpen_Auction,Open,Success
...
39600000,2,ABC,Open,Open,Failure
...
43200000,2,ABC,Open,Intraday_Auction,Success
...
57600005,2,ABC,Closing_Auction,Closed,Success
57600010,2,XYZ,Closing_Auction,Closed,Success
57600012,2,XYZ,Closed,Closed,Failure

Please submit your source code and the amount of time you spent. We will want to
compile your source code so please provide instructions for doing so.
```

Please see my `dev` *folder for solution*