2019 Spring Embedded Operating System

Homework 2

這次作業要求是請各位製作一個演唱會訂票程式的 server。預訂票的成員可以透過 telnet 來進行連線，並進行訂票行為。詳細如下。

關於 Server

1. 要求執行時讀參數

   ./<server> <console.txt> <concert.txt> <port_num>

   Server 開啟時，要能讀進 console.txt 以及 concert.txt 並進行與 hw1 一樣的內部設定，設定時要檢查演唱會的座位的種類與數量是否合乎場館的限制。若不符合規定，則印出哪一場演唱會不符合場館限制，並結束 server。
   印出**<concert_name> setting failure** 若符合規定，開啟一個 socket server listen <port_num> 的 port。

2. 要能多個 **clients** 同時進行連線
   同時在線上 Clients 數量不會超過 30 名(最多 30 名)。

3. 輸入輸出規則

   使用者端會輸出給 server 的指令為: (每次指令長度不會超過 10000 bytes) 每次送出指令後，會等到 server 回傳資訊，才會再送出下一筆指令。

   1. **exit** 將與此 client 的連線給斷掉。

   2. **<concert_name>/<ticket_class>/<ticket_number>(**空格**)<concer ……**
      本次想定的演場會的名稱，以及票種，還有數量。
      若其中有一段指令所指的演場會名稱不存在，或是演唱會的票種不存在，則輸出給使用者端
      <concert_name> not found
      或

<concert_name> does not have <ticket_class> ticket

若都符合規定，但其中一段指令的預定的票數不夠的情況下，則取消當次所有預定的票，並回傳給使用者端

sorry, remaining ticket number not enough

若每段預定的票數都<=剩餘票數時，則將訂到的票的資訊以

(以下為一行)

<concert_name> <ticket_class> <ticket_id> <book_waiting_time>

(以上為一行)

以每張一行回傳給使用者。關於**<ticket_id>**的規定是，票種加上從 0 開始的編號。

關於**<book_waiting_time>**的規定則是，從收到使用者端的指令開始計算，到要送出這張票資訊給使用者的時間為止的差距(秒)。

例如 票種名稱為 **Normal** 數量為 **2**，在 10:10:10 得到使用者的指令 ，在 10:10:40 要回傳票的資訊時

則回傳 **<concert_name> Normal Normal0 30** 下一張如果是在 10:10:41 秒回傳時

則回傳 **<concert_name> Normal Normal1 31**

輸出的票資訊的順序為，當初輸入指令的順序

3. **showall**

   當 client 輸入 showall 時，要把目前所有演唱會的資訊，照著 concert.txt 的順序輸出給使用者端。

   (以下為一行)

   <concert_name> <ticket_class_1>

   <ticket_class_1_remaining_ticket_number> <ticket_class_2>

   <ticket_class_2_remaining_ticker_number> …

   (以上為一行)

   EX : 五月天 貴賓席 500 一般席 500

4. **show <concert_name>**

   只要輸出當場演唱會的資訊就好，若演唱會名稱不存在，則輸出給使用者端

   <concert_name> not found

若獲得指令 不符合上述任何一個，則回傳給 client 端

input format not valid