Design and Programming Assignment

Operating Systems

Submit your results as a ZIP format file – no RAR.

This is an open book assignment and you may use the course materials and peer reviewed sources. However, this excludes the use of any other people and all materials that are not peer reviewed. You must cite all references.

Description

For this problem you must use Java's semaphore class. Your implementation for this problem must only use the semaphore methods to control the concurrency of your solution (IE. acquire and release).

Consider a small take-out food restaurant called Burrito Brothers that is open 24 hours, seven days a week. This very popular establishment offers a VERY-TASTY burrito. There are three servers that own a private beef area, a cheese area, and a tortilla area (IE. Each server has their own ingredients in an infinite supply). Additionally, there are three shared counter locations, a shared cash register and a shared waiting area that can accommodate up to 15 customers. By law the shop has a maximum customer capacity of 15.

A customer cannot enter the shop if it is filled to capacity. If the there is room a customer will gain access to the shop. Each customer will enter the shop with an order of one to 25 burritos. As soon as a server is free, the customer that has the shortest order is served next.

A server is either servicing a customer or waiting. Each server will make (at most) three burritos at time for a given customer. Once a server has obtained all ingredients, a burrito can be made. When a customer's entire order is finished, the customer pays a cashier and leaves the shop. Since there is only one cash register, only one patron may pay at a time. However, in the event that a customer's entire order has not been filled by the server at the completion of the current counter visit the customer must reenter the waiting area. The waiting area is organized by the shortest order next.

Implement a solution to this problem as a Java application using the Java semaphore class for synchronization. Your zip file must expand into a single directory and your application must compile at the command prompt using javac *.java. Output must include the arrival of each customer and each state transition: leaving full shop, entering shop with an order of m burritos, customer standing, getting service from server n, paying, and leaving. Each customer is represented by a corresponding thread and each server in your implementation is also represented by a unique Java thread. All customers and servers exist within a single application.

The finished assignment must include the following (submitted to Blackboard as a ZIP file that extracts to a single level directory for this problem):

- A detailed design document fully describing your work. This must be a clear and concise document that rigorously addresses the decisions and design elements that support your solution to this concurrency and synchronization problem. See the *Methodology, Design, and Writing* link in *Modules* in *Canvas* for design elements. You must pay close attention to maintaining a consistent flow in form. The document must be well written and should be proofread for detail and accuracy. Use double spacing throughout the document and the font should be Times Roman or a similar serif typeface. Use citation to provide evidence of your analysis. Citations and references should be in APA format. Use 12 point for the body of the text.

- The Java Burrito Brothers Implementation
  - The Java Source Code
  - Class files
  - The system must compile using the command: javac *.java. The system must run using the command: java Burrito <optional parameters>.
  - Output from a run of your application.