**Lab 6 Report**

*How It Works*

Data structures and Variables:

**Customers Served:**

* Represents how many customers has been served by the barber
* Related mutex locked when it is accessed

**Customer**

* Represents a customer
* Contains
  + A mutex
  + How long it will take to execute
  + A counter to track execution time

**WaitQueue**

* A queue that contains **Customer** objects
* Related mutex locked when it is accessed

Threads:

**1. Main**

* Initializes mutexes, condition variables
* Initializes waiting queue
* Creates **Barber** and **Customer Generator** threads
* Waits for those threads to exit

**2. Customer Generator**

* Generates customer using a random number generator
* When a customer arrives, checks waiting queue to see if it’s full
* If it’s not full, a new customer is created
* To create a customer, a **Customer** object is initialized, it’s mutex will be locked and a new thread is created it
* The mutex is locked to prevent that thread from executing
* The created **Customer** is then pushed to the waiting queue, and then unlocks waiting queue
* Wakes up barber if it’s sleeping
* Terminates if customers served equals defined limit

**3. Barber**

* Associated with two condition variables
  + wake\_barber
    - Wakes up barber
  + finished\_serving\_customer
    - Signals to barber that customer thread has finished executing
* Sleeps if no customers are being served and waiting queue is empty
* Waken up by **Customer Generator** if new customer arrives
* Serves customers by:
  + Popping a **Customer** from queue
  + Unlocking the mutex of the **Customer**, allowing its related thread to run and waits for it to finish
  + Once finished, increments customers served and starts over
* Terminates if customers served equals defined limit

**4. Customer**

* Contains a thread created by **Customer Generator** for it
* Contains a mutex locked by default to prevent thread from executing
* Mutex is unlocked by **Barber** once it is being served
* Increments an internal counter until it reaches defined limit
* Signals to **Barber** that it has completed, and then exits

*Running the program*

Execute the following commands:

make clean all

./barber

*Screenshots*

Text

Description automatically generated with medium confidence