Lab 5 : IPC mechanism using semaphores

October 5, 2015

Objective :

- Lab 5 is intended to learn IPC mechanism by solving the sleeping barber problem. The lab requires knowledge of threads and semaphore creation and usage. A pre-requisite exercise (Synchronizing Threads with POSIX Semaphores) has been provided (link provided on the course website) so that you learn how to create and work with semaphores. You are expected to complete the pre-requisite before starting Lab 5. You are also provided with three c-files: shm.c (describes how to use shared memory), sem.c (using semaphores) and producer_consumer.c (solution to producer consumer problem). You are expected to run these files and study them before starting Lab5.

Problem Statement

- This classical semaphore problem takes place in a barber shop. The shop has one barber, one barber chair, and n chairs for waiting customers, if any, to sit on. If there are no customers present, the barber sits down in the barber chair and falls asleep. When a customer arrives, he has to wake up the sleeping barber. If additional custom ers arrive while the barber is cutting a customer's hair, they either sit down (if there are empty chairs) or leave the shop (if all chairs are full). The problem is to program the barber and the customers without getting into race conditions.

  Implementation: You can use three semaphores:

  - ustomer, which counts waiting customers (excluding the customer in the barber chair, who is not waiting),
  - barber, the number of barbers (0 or 1) who are idle, waiting for customers, and
  - mutex, which is used for mutual exclusion.

  In the solution, a customer entering the shop has to co unt the number of waiting customers. If it is less than the number of chairs (take as input), he stays; otherwise, he leaves. When the barber opens the shop in the morning, he executes the procedure barber, causing him to block on the semaphore customers because it is initially 0. The barber then goes to sleep. He stays asleep until the first customer shows up. When a customer arrives, Barber serves the customer, starting by acquiring mutex to enter a critical region. If another customer enters shortly thereafter, he will not be able to do anything until the first one has released mutex. The customer then checks to see if the number of waiting customers is less than thenumber of chairs. If not, he releases mutex and leaves without a haircut. If there is an available chair, the customer increments an integer variable, waiting. Then he does an Up on the semaphore customers, thus waking up the barber. When the customer releases mutex, the barber grabs it, does some housekeeping, and begins the haircut. When the haircut is over, the customer exits the procedure and leaves the shop. Nobody is allowed to cut more than once.