Lab #4: Implementing Pipes: The Sleeping Barber

Due Date:

November 9, 2016 2355 hours

Implement the sleeping barber problem with simultaneous processes, random delays, and semaphores implemented with pipes.

Sleeping Barber Problem (barbers: 1, customers: 10, seats: 3)

- **Each arriving customer wakes up the sleeping barber**
- **If there are no chairs in the waiting room for the customer, the customer leaves**
- **When the barber finishes, he checks the waiting room and gets the next customer**
- **If no customers are waiting, he goes to sleep**

The outline of a program has been provided implementing a number of elements of this program. Your task is to implement customer_run() and barber_run(). This program will be far easier if you write the pseudocode for a solution before working on any code.

Program requirements:

1. The barber_run() should use pipes in several different ways:
   - **seats** as a mutex in order to modify the value of freeseats
   - **barber** as the semaphore for customers to wait for the barber (but only after sitting in the waiting room)
   - **customer** as the semaphore for the barber to wait for customers
   - **freeseats** as a pipe holding the integer value of the number of seats in the waiting room

2. The customer_run() will need code for 2 possible cases: when there are chairs in the waiting room and when there are not chairs in the waiting room
3. Pipes are communication devices. To use a pipe to store a variable, you must first initialize it by writing a value to the pipe. To modify the value, first read from the pipe which removes the value, change the value, and then write it back to the pipe. Doing all the steps requires exclusive access.
4. 10% Extra Credit: The barber_run code executes 10 times because at most 10 customers may have their hair cut. In practice, you cannot predict how many of the 10 customers will turn away. Is it possible to know how many exact times the barber will need to run before it will never find another customer? Is it possible to clean up all child processes perfectly and print a Done! statement? Test with ps.
5. **5% Extra Credit: instead of Moodle, submit your assignment via https://education.github.com/ (MUST be private, invite swirsz) Make sure your full name is included in the comments. I'll send an email to all individuals who I have shared Github access 1 day before the assignment is due informing them I will retrieve their program from Github.**

You must submit **two** things: program files and the output from running the program. Sample output on the back of this handout.

```
bash-4.1$./program5
- New customer trying to find a seat
- Customer is decreasing the number of free seats to 2
- Customer is now waiting for the barber
- New customer trying to find a seat
- Customer is decreasing the number of free seats to 1
- Customer is now waiting for the barber
- New customer trying to find a seat
- Customer is decreasing the number of free seats to 0
- Customer is now waiting for the barber
- New customer trying to find a seat
* Customer giving up: No free chairs in waiting room
Barber 1 is trying to get a customer
Barber 1 is waiting for the seat to become free
Barber 1 is increasing the number of free seats to 1
Barber is now cutting hair 1
- Customer is now getting a hair cut
- New customer trying to find a seat
- Customer is decreasing the number of free seats to 0
- Customer is now waiting for the barber
- New customer trying to find a seat
* Customer giving up: No free chairs in waiting room
- New customer trying to find a seat
* Customer giving up: No free chairs in waiting room
Barber 2 is trying to get a customer
Barber 2 is waiting for the seat to become free
Barber 2 is increasing the number of free seats to 1
Barber is now cutting hair 2
- Customer is now getting a hair cut
- New customer trying to find a seat
- Customer is decreasing the number of free seats to 0
- Customer is now waiting for the barber
- New customer trying to find a seat
* Customer giving up: No free chairs in waiting room
- New customer trying to find a seat
* Customer giving up: No free chairs in waiting room
Barber 3 is trying to get a customer
Barber 3 is waiting for the seat to become free
Barber 3 is increasing the number of free seats to 1
Barber is now cutting hair 3
- Customer is now getting a hair cut
Barber 4 is trying to get a customer
Barber 4 is waiting for the seat to become free
Barber 4 is increasing the number of free seats to 2
Barber is now cutting hair 4
- Customer is now getting a hair cut
Barber 5 is trying to get a customer
Barber 5 is waiting for the seat to become free
Barber 5 is increasing the number of free seats to 3
Barber is now cutting hair 5
- Customer is now getting a hair cut
Barber 6 is trying to get a customer
```