CS 375 - UNIX System Programming
Fall 2015 – Programming Project 6
30 points

Out: November 3, 2015
Due: November 12, 2015 (Thursday)

Assignment

Write three programs that communicate using signals and **POSIX** (not System V) IPC methods. The three program files should be named **`master.cpp`**, **`reverse.cpp`** and **`upper.cpp`**. One message queue, one shared memory segment and two semaphores are required. The master program should prompt the user for a line of input. The master program should pass the user input to the reverse program via a message queue. The reverse program should reverse the line of input and place it in a shared memory segment for further processing by the upper program. The upper program should convert the text in the shared memory segment to upper case and then send it to standard output.

The message queue provides the necessary synchronization mechanism between the master and reverse programs. Use a pair of named semaphores to provide for exclusive access (and synchronization) to the data shared between the reverse and upper programs. The upper program should send a signal (SIGUSR1) to the master program after it has sent the converted text to standard output. On reception of the signal, the master program should prompt the user for another line of input. You may assume that the longest possible line of user input is 256 characters.

The master program should create the IPC object (message queue, shared memory segment, semaphors). The master program also is responsible for forking and exec'ing the reverse and upper programs. The reverse program should take the message queue, shared memory segment, and semaphore names as its arguments (four total). The upper program should accept the shared memory segment and semaphore names as its arguments (three total).

When the master program reads end-of-file on standard input (user types ctrl-d), it should send a shutdown message (via the message queue) to the reverse program. Before the reverse program exits it should send a shutdown message (via the shared memory segment) to the upper program. The master program should wait for both the reverse and upper programs to exit, before cleaning up and exiting. No IPC objects should remain after the master program exits.

Example session is shown below (user input is shown in **bold**):

```
$ ./master
> Hello there!
!EREHT OLLEH
> How are you?
?UOY ERA WOH
> ^D
$
```

Note: as in Project 5, each invocation of the master program should fork and exec the two children, the reverse and upper programs, only once.

What to submit

- Provide a makefile named **`Makefile`** that will make all three programs for this assignment as the default target (typically called **`all`**). Each program must be a separate target.
- Create a tarfile or zipfile containing your program source files and makefile.
- Submit your archive using the submission system (http://submission.evansville.edu). The grading script only will make the project and check that executables named **`master`**, **`reverse`**, and **`upper`** are produced. It will not run anything.