Shubham Sharma  
CprE 308: Section L  
Project 02

**IPCS**

**Unnamed Pipe**

**1pt) Output of pipe\_test.c**

****

**1pt) What do you notice about the timing of the printing?**

There was about a 2 second delay before the printing of statements.

**2pt) What happens when more than one process tries to write to a pipe at the same time? Be specific: using the number of bytes that each might be trying to write and how that effects what happens.**

Answer here

**1pt) How does the output of pipe\_test.c change if you move the sleep statement from the child process before the fgets of the parent?**

The output after moving the sleep statement.

This seems to be no difference in the output compared to the original code. (According to the lab computers)

**1pt) What is the maximum size of a pipe in linux since kernel 2.6.11?**

The max size of a pipe is: 1048576 Bytes. (according to the /proc/sys/fs/pipe-max-size)

**Named Pipe (FIFO)**

**1pt) What happens when you run the echo command?**



The echo text is written into the test\_fifo pipe, once finished the echo command will block the pipe from anything else to be read.

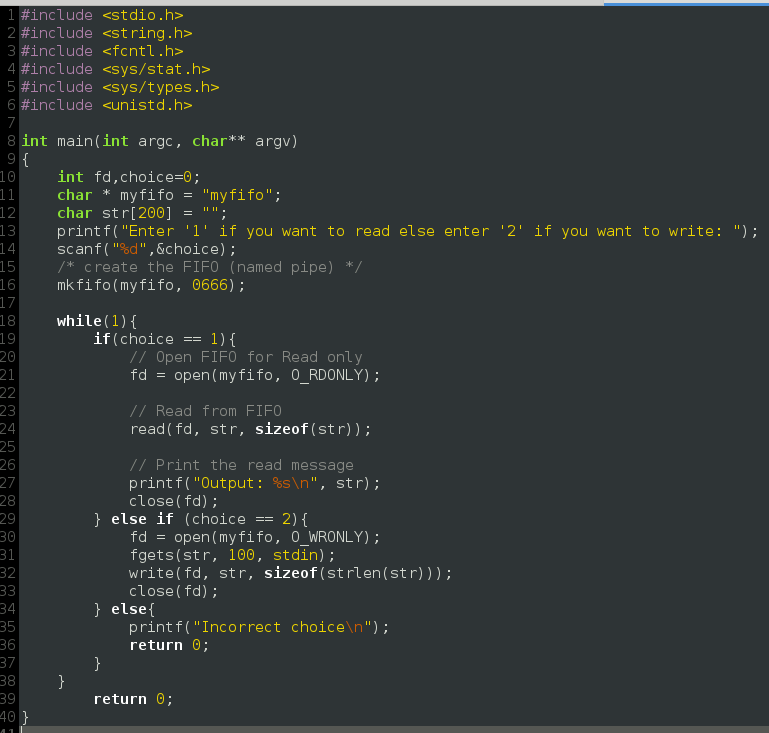
**1pt) What happens when you run the echo first and then the cat?**

Here, the echo writes into the pipe and then the cat command will read the text in the pipe blocking anything else to be written.

**2pt) Look at the man page fifo(7). Where is the data that is sent through the FIFO stored?**

According to the man page, when processes are exchanging data via the FIFO, the kernel passes all data internally without writing it to the filesystem. (The data is sent is stored in the kernel)

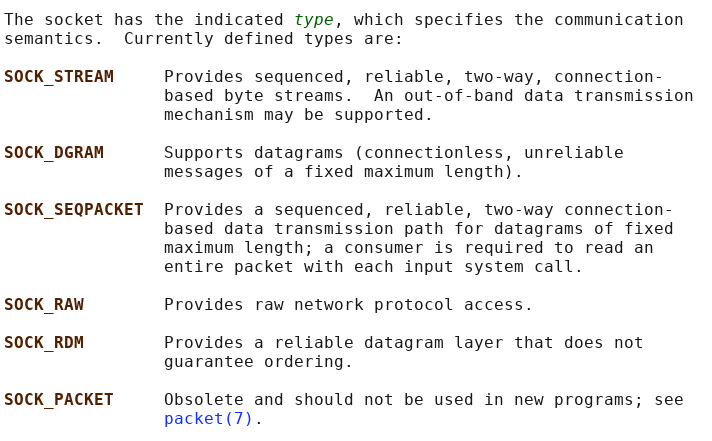
**2pt) Write a short program that uses named FIFO (mkfifo(3)) to print any line entered into the program on one terminal out on the other terminal.**



**Socket**

**2pt) What are the six types of sockets?**

According to the [man pages](http://man7.org/linux/man-pages/man2/socket.2.html), the size types of sockets are as follows:



**1pt) What are the two domains that can be used for local communications?**

The two domains that can be used for local communications are AF\_UNIX, AF\_LOCAL.

**Message Queues**

**1pt) What is the output of mq\_test1?**



mq\_test1 prints the messages from mq\_test2

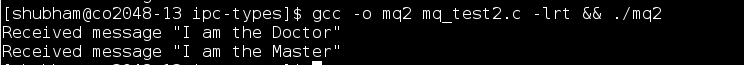
**1pt) What is the output of mq\_test2?**



mq\_test2 prints the messages from mq\_test1

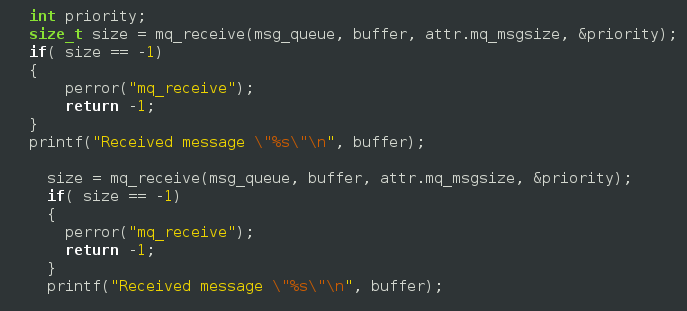
**2pt) Change mq\_test2.c to send a second message which reads “I am X” where “X” is your favorite companion. Change mq\_test1.c to wait for and print this second message before exiting.**



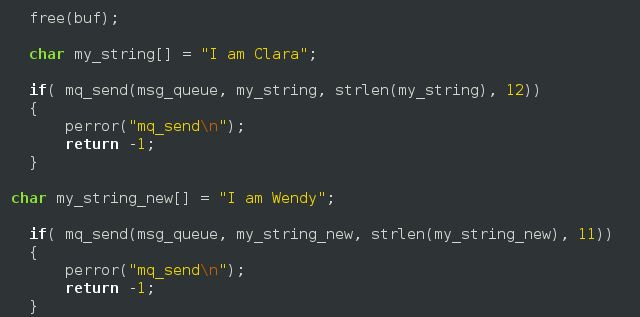


Here we can see that mq1 prints the companion name second. This was down by modifying mq1 and mq2 as follows:

mq\_test1:

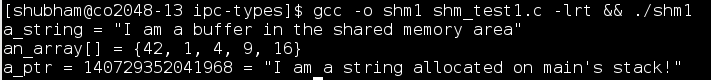


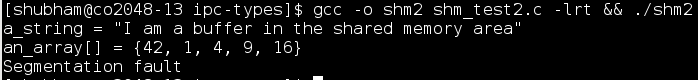
mq\_test2:



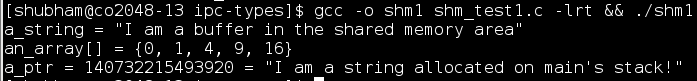
**Shared Memory Space**

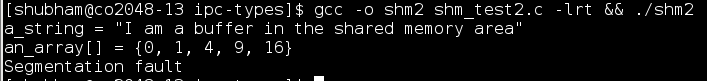
**1pt) What is the output if you run both at the same time calling shm\_test1 first?**



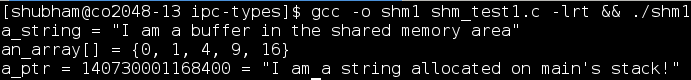


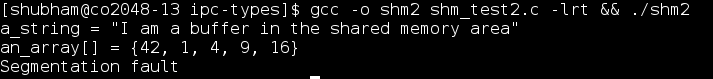
**1pt) What is the output if you run both at the same time calling shm\_test2 first?**





**1pt) What if you run each by themselves?**





**2pt) Why is shm\_test2 causing a segfault? How could this be fixed?**

The segfault is caused when printing string value of shared\_mem→a\_ptr. This is because the structure isn't self-contained. The structure may be within the shared memory but the pointer cards can point anywhere.([source](https://stackoverflow.com/questions/26564854/segmentation-fault-when-accessing-shared-memory-segment)).

This can be fixed by allocating a size to the pointer since it is a dangling pointer.

**1pt) What happens if the two applications both try to read and set a variable at the same time?**

If two applications both try to read and set a variable at the same time, then there exists a race condition. The output of this system is unpredictable and should always be avoided.

**1pt) How can a shared memory space be deleted from the system?**

This can be done by using the shm\_unlink command.

**2pt) Change the code to share some useful piece of information?**

Code here

**Unnamed Semaphores**

**2pt) What is the function call that would be needed to create an unnamed semaphore in a shared memory space called shared\_mem->my\_sem and assign it an intial value of 5?**

Answer here

**Named Semaphores**

**1pt) How long do semaphores last in the kernel?**

Answer here

**1pt) What causes them to be destroyed?**

Answer here

**2pt) What is the basic process for creating and using named semaphores? (List the functions that would need to be called, and their order).**

Answer here

**Signals**

**1pt) What happens when you try to use CTRL+C to break out of the infinite loop?**

Answer here

**1pt) What is the signal number that CTRL+C sends?**

Answer here

**1pt) When a process forks, does the child still use the same signal handler?**

Answer here

**1pt) How about during a exec call?**

Answer here

**5pt) Write two programs. One which will send a signal of number 42 to the other process. The other program should catch that signal and print out the message “I got the signal!”**

Code here

**Dynamically / Statically Linked Libraries**

**1pt) First output of lib\_test:**

Answer here

**1pt) Second output of lib\_test after exporting the library:**

Answer here

**Project 2**

**If you worked with someone else – who was it?**

Name here

**5pt) Summary**

Answer here

**If you did extra credit – tell us what the functionality and how to use it here:**

Answer here

**How to run Project 2**

**Terminal 1: ./src/printer-server/printer**

rm /drivers/\*

make clean

make

./virt-printer –n printer0

./virt-printer –n printer3

**Terminal 2: ./src/printer-server**

make clean

make

./main -d

**Terminal 3: ./src/libprintserver**

make clean

make

**Terminal 4: ./src/cli-printer**

make clean

make

export LD\_LIBRARY\_PATH="../libprintserver/"

LIST FUNCTUION:

./cli-printer –l file\_name

PRINT FUNCTION

./cli-printer –d driver –s description –o output\_name file\_name

**Notes:**

Any other details needed to run the program