Program 4

Due Date: December 2, 2008 **Points:** 100

Program

1. (*100 points*) The MINIX kernel does not implement user-level semaphores. Your task in this project is to do so. The semaphore system calls have the following interface:

```
#include <minix/semaphore.h>
int semup(int semaphorenumber);
int semdown(int semaphorenumber);
int seminit(int semaphorenumber, int value);
int semdestroy(int semaphorenumber);
int semstatus(int semaphorenumber, int *value, int *numblocked);
```

   In all the above system calls, `semaphorenumber` is the number of the semaphore (0 through 9). In *`seminit`*, the initial value of the semaphore is set to `value`. In *`semstatus`*, the current value of the semaphore is returned in `value`, and the number of processes blocked on that semaphore is returned in `numblocked`.

   The system call *`seminit`* creates the semaphore and returns 0 if the semaphore does not exist. The initial value is set to `value`. It returns -1 if the semaphore exists. The system call *`semdestroy`* deletes the given semaphore and returns 0. If any processes are blocked on it, though, it returns -1 and does not delete the semaphore. All other calls return 0 on success and -1 on failure.

   Please implement the producer-consumer problem to verify your semaphores work correctly. Your program must use the semaphores to synchronize accesses to the bounded buffer (to be implemented as a file) and the necessary counters (also kept in a file). The program will open and initialize the files, initialize the semaphores, and then fork a specified number of producer and consumer processes which produce or consume a specified number of items.

   Hint: Implement these in the memory manager (MM), where tasks can be easily blocked and restarted. Note that you do not need a synchronization mechanism within the kernel to protect the semaphore structures because while the memory manager is running, no other process will run..

Extra Credit

1. (*20 points*) Implement appropriate `errno` codes and *`perror`* strings to describe errors in the above system calls in more detail.

   Write a program to demonstrate these work.