Laboratory 2

Due Date : May 4, 1999 **Points** : 100

These problems ask you to make small modifications to the MINIX kernel. You will be modifying the kernel's source code, recompiling and assembling it, using *make*(1) to link the various executable files, and then produce a new MINIX boot "disk."

These programs require only rather minor modifications to MINIX. The key is locating the current code that should be modified; surely, there are many acceptable solutions.

To receive credit for your solutions, show the TAs the modifications you have made to the kernel source code. Also demonstrate your new kernel running programs that show that your new kernel meets the requirements of the problem. Think carefully to come up with programs that demonstrate your new kernel.

Submit a boot disk with your new Minix system to the homework box by the deadline. Put your sources for your changes to the MINIX kernel, and the program(s) you wish to run to demonstrate your new kernel, onto a second disk, and submit that disk as well.

1. Modify the MINIX kernel to record the number of *open, close, read, write, exec, getpid, fork, stat, unlink, kill* system calls each process makes. That is, you will use an array, perhaps as part of the process table, to record for each process how many times it has made a specific system call, *e.g.*, *open*. When a particular hot-key is depressed (you pick the key), this array will be displayed. To convince yourself that the kernel is functioning according to the requirement, run a program that will make lots of system calls and show that they are being recorded. Remember to initialize the array when the process begins!

   For details on the hot keys, look over the description on the bottom of page 295.

2. Modify the MINIX process table so that it contains a *priority* field for each process. The priority of a newly created process is to be zero, and the priority is to decrease by 1 each time the process loses the CPU — whether by having its quantum expire or when it makes a system call. When the scheduler picks a process to run, it will pick the highest-priority process.

   Pick a hot-key that when struck will print the priorities of all processes.

   You will be asked to demonstrate that your new MINIX kernel properly modifies priority and always schedules processes with highest priority. To do this, you might want to have a number of long processes around, and then create a few short processes and demonstrate that these get scheduled as you would expect.