Homework #2

Create a simple client/server system in C that is capable of transferring files between two different machines.

When a client is invoked with a specific server successfully, it will quietly wait for commands. A set of at least four commands is available for the user to select. They are shown below with the allowed options:

send ***filename*** **[*filename*]**
recv ***filename*** **[*filename*]**
help
quit

where

send and **recv** are for sending and receiving a file respectively.
filename indicates the source filename, and **[*filename*]** indicates an optional target filename. If the second argument is missing, the source and target have the same filename.
help displays these four options.
quit quits the program

When the server is invoked, it simply sits there waiting for client requests. However, it is not a silent server. It should print out status information on its console such as:

client requests connection.
client connection request is granted.
client requests ***filename***.
client request for ***filename*** is sent successfully.
client terminates.

Replace ***client*** and ***filename*** with the actual client name and filename involved. It is a concurrent server. You must use **fork()** in your server program to handle multiple client requests concurrently. For simplicity, all files are handled the same way: No distinction between binary and ASCII files. Adequate error checking is required. Examples include wrong server name, file requested dose not exist, wrong command, etc. Also, your server program must reject any port number which is not in the range of 50,000-60,000 until a valid one is provided when the server starts.

In addition, signal processing is also required in this assignment.

a) On the server side, the parent must wait for its child processes to terminate. To be specific, the parent needs to catch the SIGCHLD signal when a child process finishes execution. The termination message must be printed by the signal handler invoked in response to the termination of the child.

b) On the client side, the client process will ignore the SIGINT signal once a file transfer has started.

You may use either signal() or sigaction() in this assignment. However, sigaction() is preferred. In addition, there will be no menu-driven option. You must follow the syntax specifications exactly.

Name your files based on the functionality of the programs: i.e., **hw2Ser.c**, **hw2Cli.c**, etc.

a) The syntax for invoking the server is: **hw2Ser** ***valid-port-number***

b) The syntax for invoking a client is: **hw2Cli** ***server-address server-port-number***

Submission of your files must be done electronically via the instructor's Web page or by accessing the URL ***http://www.cse.fau.edu/~hwsam/student/index.html*** directly. Follow the screen prompts. As a reminder, no email submission will be accepted.

This is an individual programming assignment. Group discussion is allowed. However, each student must code his/her own program independently.

Due dates:

1. Students taking *live class* please select the **Live-class** option. The due date is October 26, 2010 before 11:30 pm.

2. Distance learning students watching lecture videos from remote sites please select the **Remote-FEEDS** option. The due date is October 28, 2010 before 11:30 pm.

Note: You are encouraged to use C to do your homework assignments in this course. However, other programming languages are also allowed if you feel more comfortable that way. In this case, you still need to submit all your source files to HWSAM, along with a readme file detailing how to run your programs. In case I don’t have the software package you have used, I may ask you to bring your system to my office for run test purposes.