San José State University
Department of Computer Engineering

CMPE 142
Operating Systems
Section 1

Spring 2021
Instructor: Ron Mak

Assignment #9

Assigned: Friday, April 23
Due: Friday, April 30 at 11:59 PM
Team assignment, 100 points

POSIX `select()` and socket connections

In this assignment, your team will make a socket connection to a remote server process and call the POSIX **`select()`** function to monitor the connection and other input sources.

The remote time server

Make a small change to the time server program given in class that accepts socket connections from client processes and sends the current time to the connection. The program uses port 5000 for the socket, but you can choose another number that is greater than or equal to 1024.

Whenever it accepts a connection, the time server process should also print to its standard out each time message that it sent to the connection.

The change you should make to the program is to have the server sleep randomly for 1, 2, 3, 4, or 5 seconds after each time it sends the time.

The client program

Your client program should accept input from several input sources, including the time server. Input will come via file descriptors that the POSIX **`select()`** function will monitor.

In the client program, create two pipes and fork two child processes.

Child process #1 writes **`Message 1`**, **`Message 2`**, **`Message 3`**, etc. to the write end of pipe #1. Between writes, it should sleep randomly for 1, 2, 3, 4, or 5 seconds.

Child process #2 repeatedly prompts the user (you) to type a line of text at the keyboard, which the child process then reads and writes to the write end of pipe #2.

In an infinite loop, the parent process should use a **`select()`** call to monitor the read ends of both pipes (two input file descriptors) and the socket connection (another input file descriptor) to the time server. Use a 10-second timeout. Whenever an input file descriptor becomes ready, the process should read and print the data to its standard out. Label the data with its source: child #1, child #2, or time server.

Tips

You must reinitialize the input set with the input file descriptors each time before calling **`select()`**, because that function call modifies the input set.

Because the time server process sends one time message for each socket connection, the client process needs to close the socket connection and make a new connection after receiving time data.

The time messages that the time server process prints and the time data that the client program receives should match according to their respective standard outs. In other words, there shouldn't be any missing time messages in the client program's standard out.

Instead of forking the second child, you can have the parent process monitor its standard in directly to detect that the user has typed a line.

What to submit

Submit the following to Canvas, **Assignment #9: POSIX `select()` and Socket Connections**.

- Source files (either C or C++) of your time server and client programs.
- A text file of a few dozen lines of the time server program's standard out.
- A text file of a few dozen lines of the client program's standard out that clearly shows reading and printing data from all three input sources. These lines should include the time messages in the previous text file to demonstrate that no time messages were lost.

Rubric

- Your submission will be graded according to these criteria:

Criteria	Max points
Client program	**80**
• Create pipe(s) and child process(es).	• 10
• Child #1 writes messages to its pipe.	• 10
• Child #2 prompts the user for text lines and writes the lines to its pipe (or this can be done by the parent process).	• 10
• Create a socket connection to the time server process.	• 15
• Successfully use **`select()`** to monitor the input sources.	• 25
• Sample input data printed to standard out.	• 10
Time server program	**20**
• Create a socket.	• 5
• Wait for connections to the socket.	• 5
• Accept a connection and send a time message, followed by a random sleep.	• 5
• Sample printed time messages to standard out.	• 5