**GO THROUGH THIS LIST BEFORE TURN IT IN!!!**

**Command:**

* SERVER:
  + \list
  + \kick
  + \exit
  + <any-other-text>
* USER:
  + \list
  + \exit
  + \p2p
  + <any-other-text>

**REQUIERMENTS:**

* SERVER:
  + Polls STDIN and set of open PIPES in a **LOOP**
  + **LOOP** terminate when the server’s exit command is used on **server** process
  + Use **usleep**, NOT **sleep()**
* USER:
  + Polls STDIN and PIPES from the child
  + **usleep()**

**IMPORTANT!!!**

* Use **kill()** to kill a process
* Close ends of pipe that does not need
* Remove new line (\n) when read from STDIN
* Read() will return 0 if the pipe is closed (broken). Check this return value to detect if the user or server processes are failed (terminated). **YOU MUST CLOSE PIPES APPROPRIATELY (NOT USED FOR A PROCESS) TO DETECT IF THE PIPE IS VALID OR NOT.**
* If the user connects with a name already used, the connection will be closed.
* Check the return value of all system calls!!. Print out the error. If any error prevents your program from functioning normally, then it should exit after printing the error message.
* Before execute **\exit** on the SERVER, the main SERVER process and its child processes **must exit properly**, **cleaning up** all of the user, **waiting** for all child processes and freeing up any used resources.
* Use **usleep()** in all polling loops
* You might need to build up strings for message, use (**sprintf**)
* Strcpy(), strncpy(), strtok(), strlen() may be handy

**NOTE:**

* Users may join a leave the chat, or their chat code may fail unpredictably, but the chat service should keep running until the chat server decides otherwise.
* The USER process must display the name of the user as part of the prompt

**ERROR CHECKING:**

* Check for unexpected failure such as Control C (in user or server), user close the terminal.
* A crashed user should be cleaned up and not distrupt the SERVER.
* A crashed server should allow every USER to terminate automatically (Think about detecting failure via pipes)
* Take care of zombie or/and orphaned processes.
* Add another command: \seg – create a segmentation fault in the user process by any means (e.g. char \*n = NULL; \*n = 1;). The result of this should be that the SERVER cleans up that user and otherwise the chat should run smoothly.

**READ THE GRADING CRITERIA FOR THE PROJECT!**