simple_daemon.c accepts multiple connections by using the TCP protocol to establish connections. When a new connection is established it forks a child process to handle.

When sending SIGINT it is sent to the underlying telnet daemon not the local telnet process. Therefore, it is up to the daemon to handle the signal. In this instance, the client intercepts the signal and does not pass it to the daemon. It treats it as an ignored command.

When connecting to the daemon with multiple clients the daemon forks child processes to handle the connections. There is no overlap between the shell windows because they are handled by different threads and processes.

SIGHUP, SIGTERM, SIGINT as long as they are sent to the parent daemon process.
5.
When the `shell` receives the 'quit' command it breaks its while loop, returns 0 and then exits. The daemon is meanwhile watching the socket to see if the client sends any information, when it sees that the client has sent information it then sees what return value was returned from the shell. If it is 0, that means the client has terminated the connection so the daemon then terminates the connection.

6.
When the daemon receives the SIGCHLD signal it knows that a child process has terminated. The child_terminate() function is set to handle the SIGCHLD function. When it is called, it retrieves the process ID of the child process and closes/clears the its data socket. kill_child() is used when a child process recieves SIGTERM from the parent process.

7.
In this case the parent process is `/usr/sbin/sshd`. When a client connects it spawns a process to authenticate the user. After the user is authenticated, another process will be created that has higher privileges. This is privilege separation that is used to prevent unauthorized privilege escalation.