Task

In your shell, you can use `kill -XXX PID`, where `XXX` is the human-friendly suffix of the desired signal, to send any signal to the process with process id PID. For example, `kill -TERM PID` sends a `SIGTERM` to the process with process id `PID`.

In C, you can use the `sigaction` system call to change how signals are handled by the current process. The shell should basically ignore most of these signals, whereas the shell's subprocesses should respond with the default action. For example, the shell should ignore `SIGTTOU`, but the subprocesses should not.

Beware: forked processes will inherit the signal handlers of the original process. Reading `man 2 sigaction` and `man 7 signal` will provide more information. Be sure to check out the `SIG DFL` and `SIG IGN` constants. For more information on process group and terminal signaling, please go through this tutorial.

Your task is to ensure that each program you start is in its own process group. When you start a process, its process group should be placed in the foreground. Stopping signals should only affect the foregrounded program, not the backgrounded shell.
