Lab #9: signals

Overview

The purpose of this lab is to introduce the use of signals.

`ignore`

Write a program, named `ignore`, that repeatedly reads "lines" from the keyboard until the user enters `quit`. This program must prevent termination due to `^C`. Set up a signal handler to respond to such attempts to terminate the program with a reminder on the proper technique to quit.

`timeout`

Write a program, named `timeout`, that can be used to limit the duration of another program. `timeout` takes, as command-line arguments, an integer number of seconds and another command (optionally with arguments of its own). The `timeout` program must spawn a child process to execute the argument command and set an alarm that will be triggered after the specified number of seconds (use `sigaction`). If the child process has not terminated by the time that the alarm has triggered, then it should be killed (and the exit status of timeout should be non-zero). If the child process terminates, then the exit status of `timeout` should be that of the child process.

For example,

```
% timeout 5 sleep 10
Killing child ...
% echo $?
1
```

You need not worry about the case when the alarm might trigger exactly when the child process terminates (this is an example of what?).