Tutorial 9
Signals

Learning Objectives

After this tutorial, you will be able to:

- Write and use a Makefile
- Use signals to communicate between processes. Specifically:
    - Send a signal to a process
    - Install signal handlers in a process to catch incoming signals

Tutorial

In order to receive full marks, you must complete everything in this tutorial section. There are no extra exercises this week. You will be modifying this week's code in next week's tutorial.

1. Download the file `T09.zip` from the tutorial module on Brightspace. Extract and read through the tutorial files. There is also a zip (unzip) tool in Linux. Read the `man` page for the zip tool to see how to unzip the tutorial files.

2. Create a `Makefile` to compile `send.c` and `handle.c` with targets `send` and `handle`. Make `send` and `handle`.

3. Run `handle` in the background, you should see something like:

   ```
   $ ./handle &
   [1]  2329
   ```

   Where `2329` is an example of a process ID. In this case, the ID of the running `handle` process is `2329`, but it will be different for your process.

4. Run `send`, enter the PID of the `handle` process, and enter 1, 2, then 0.

5. Run `ps` to list the current running processes. Why is `handle` still running? Kill it.

   ```
   $ kill 2329
   ```

   Run `ps` to check that the process has terminated.

6. Modify `send.c` so that when the user chooses 0, a signal (`SIGINT`) is sent to the process pid.

7. Make send and re-run `handle` (again in the background), run `send` and enter 0. Run `ps` to verify that `handle` is not still running.

Note: If you have completed this tutorial early, Tutorial 10 - Processes and Threads, has been released in advance so that you can continue working with this material. You will need the files from this tutorial for T10.

Saving and Submission

In-Person Attendance: Submission is optional, as you can be checked off in class, but you must still complete the required portion of the tutorial for marks.

Asynchronous: If you are completing the work on your own at home, make sure to follow the submission requirements.

1. Create an archive to submit your files, and include all of the files that you worked with in the tutorial.
   a. They may be submitted as a .tar, .tar.gz, bz2, or .tgz file
   b. Make sure to include all of the code needed to run your program

2. For **full marks** you should have completed all problems in the Tutorial section (and they should be seen upon running the program).

3. For **part-marks** you should have attempted to complete most of the tutorial. Grades for part-marks will be at TA discretion.