EECS 3221.3
Assignment 1.

Processes, Signals and Interprocess Communication.

Due Date: May 23 at 4:00pm.

1. Exploring Linux Use the command `strace` to trace the system calls of the execution of
`wc <file>`
and answer the following questions (in a single plain ascii file):

(1) Why does your process read files like `ld.so*` when it starts up?

(2) Why does it read files like `*locale*` immediately afterwards?

(3) Program `wc` reads the file one chunk at a time. How big is this chunk?

(4) Which system call does not return anything?

2. A Program for Marking Assignments

Marking an assignment usually involves running it and comparing the data generated with the data that should be generated. Of course there are many other details involved, like the program crashing, going into an infinite loop, writing files it should not, etc. In this assignment you will develop a tool that marks an assignment that involves two programs connected by a pipe and you will restrict the execution time to maximum 3 seconds. Your tool will provide the standard input of the first program with data from file `"test.in"` and record the standard output of the second program to file `"test.out"`. The standard error of both programs will be recorded in file `"test.err1"` and `"test.err2"` respectively.

You will write a program called `marker` that accepts at least three arguments as for example:

```
    ./marker ls CourseDir -p- wc -c
```

which means that `marker` will start a process running `ls` with argument `CourseDir` and then start program `wc` with argument `-c`. It will also create a pipe and cause the output of the first program to write to the pipe and the second program to read from the other end of the pipe. The argument `"-p-"` serves as a separator. The standard input of the first program and the standard output of the second will be redirected from and to `"test.in"` and `"test.out"` respectively. The standard error from both will be redirected to `"test.err1"` and `"test.err2"`. The `alarm` system call will be used to notify `marker` that 3 seconds have gone by. If one or both processes are still running then the living child is sent a `SIGKILL` signal and an error message

```
        marker: At least one process did not finish
```

written to the standard error of the tool. For each process that finishes the tool prints to the standard output a message

```
        Process ls finished with status 2
```

or

```
        Process ls finished with status 0
```

depending of course on the return status.

2.1. Structure of the program.

Use the provided skeleton program and complete the `main`, `start_child` and `f_error`. If you need to add more functions, add them to `fork.c` if they are called from `fork.c` or to `main.c` if they are called from `main.c`.

2.2. Submitting your assignment

Use the command `submit` from any departmental computers running Linux. In particular type

```
        submit 3221 A1 fork.c main.c
```

Notice that you are submitting only two files. Avoid any modification to the other files since this may render your program uncompilable.

3. Programming Standards and Documentation

You have to write high quality code with good error checking and graceful error handling. The return values of all system calls that can return an error have to be checked, all user interaction has to be examined, no buffer overflows should be possible etc. Furthermore no freezing, crashing or deadlocking is acceptable.

The skeleton program includes a standard makefile that allows compilation, and cleanup. The manual should be plain ascii files and follow the style of man pages in section 1 about half a page long, and should be targeted to a user (like a TA) of the tool.

4. Asking Questions

Please e-mail assignment related questions to minas at eecs.... The answer will be posted on the FAQ.

5. Reading

Besides the code in the course web pages, you should read several man pages. Pages like kill(2), sigaction(2), wait(2), pipe(2), fork(2), read(2), write(2), open(2), close(2), signal(7), errno(3), dup(2), dup2(2) are good starting points.