CSC209 Fall 2016

Week 11 in-class exercise. Sending signals between parent and child

Write a program that forks two children.

One child sends a SIGUSR1 signal to the other child approximately every 2 seconds (use sleep(2) between signals.).

The other child does nothing except print out numbers from 1 to 1000. But every time a SIGUSR1 signal arrives, it prints "quit poking me" to standard error. When it's counting is finished, this child exits with the number of times it was poked as the exit code.

The parent then prints the number of pokes to the standard error.

```
#include <stdio.h>
#include <signal.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/wait.h>
```

//Declare a variable to store number of pokes.
//it must be a global variable – why?

//write a signal handler
```
void poke(int signal) {

}
```

```
int main() {
	//fork the first child
	int pid_first = fork();
	if (pid_first == 0) {
		// I am the first child who counts to 1000
		// First install signal handler for SIGUSR1
```

```
        // now count to 1000



        //Now exit with number of pokes as an exit status


    }
    //The first child counts and exits, so only parent gets here
    // I am the parent about to fork 2nd child
    int pid_second;
    if ((pid_second = fork()) == 0) {
        // I am second child who knows first child's PID:
        while (1) {
            sleep(10);
            //send SIGUSR1 signal to the first child



        }
    } else {
        // I am the parent
        int status;
        waitpid(pid_first, &status, 0);

        //extract exit status from the status variable and print



        fprintf (stderr, "Number of pokes is %d\n", _______________)

        //Send terminate signal to the second child



        exit(0);
    }
}
```