IS F462 Network Programming (Lab Exercise 1)

`$ R.VIJAY KRISHNA @ 2017A7PS0183P`

General Outline of the Program

Data Structures used

- A simple Queue Data Structure has been used to process the PIDs of the child processes and cycle through it repeatedly via the parent process. This avoids having to explicitly keep track of the PIDs of the children, and instead get them from the Queue.

Functions used

sig_usr1(int)

- This is the signal handler function for all the child processes. This handler receives the `SIGUSR1` user-defined signal and reads from the FILENAME and sleeps for 1 second. The handler completely ignores any `SIGUSR1` signal that the Parent process can produce during this time, as the process is not free. Whenever the process finishes it's cycle, it again enables the signal to be passed to it, and sends another signal `SIGUSR2` to the parent, indicating to the parent that it is now free to receive `SIGUSR1` and run the task again.

sig_usr2(int, siginfo_t*, *void**)

- This is the signal handler function for the parent. This does not have much to do, other than to update the Console that the calling child process is now free. To get the pid of the calling process, I declared it as `sigaction()`, rather than `signal()`, as the `sigaction()` API enables more functionality and portability.

The main() function

- First, the parent forks to `N` children, with PIDs ready to be pushed into the Queue.
- Each child enables the `sig_usr1` handler, as it is now ready to run.
- It then waits for a signal from the parent process to continue.
- In the parent process, create the Queue of all the children.
- Now, since the Queue is also ready, send a signal to all the PIDs in a round robin fashion using the Queue and a Circular Shifting of the Queue.

- Since the parent doesn't need to respond to `SIGUSR2`, it can continue doing the same, sending signals to the Queue. The child process will run whenever it is free, thus achieving our goal.

Reading the File

- The program reads only one line of the File, that is, it only reads characters until the first newline or EOF. To make the Output look cleaner, I have assumed this in the program, and since the problem statement states that only one line of the file is read.

Compiling and Running the Program

- For compiling and running the program, invoke with

```
make run FILENAME NO_OF_CHILDREN
```

- For example :

```
make run sample.txt 5
```

- The above would compile and run the program that reads from the file "sample.txt" and the Number of Children (N) = 5.