

CST8234 – C Programming F17 (Lab 7)

Programming Exercise

In this lab, we will gain experience with

- Set up bi-directional communications between two processes
- Work with file descriptors, and blocking vs non-blocking reads

Statement of the problem

You'll create three files: main.c, parent.c, and child.c.

You will implement 'main()', such that it will

1. Set up two pipes, then fork your process, then close down the unused halves of each pipe, as per the lecture notes on "Process Control (IPC).
2. Then call either 'void runAsParent(int fdRead, int fdWrite);' or 'void runAsChild(int fdRead, int fdWrite);' depending on whether the return value of 'fork();' indicates you are the parent or the child.

You must do all the appropriate error checking to make sure that your pipes were created and that the fork() succeeded. Before exiting, you'll also make sure that you close all file descriptors.

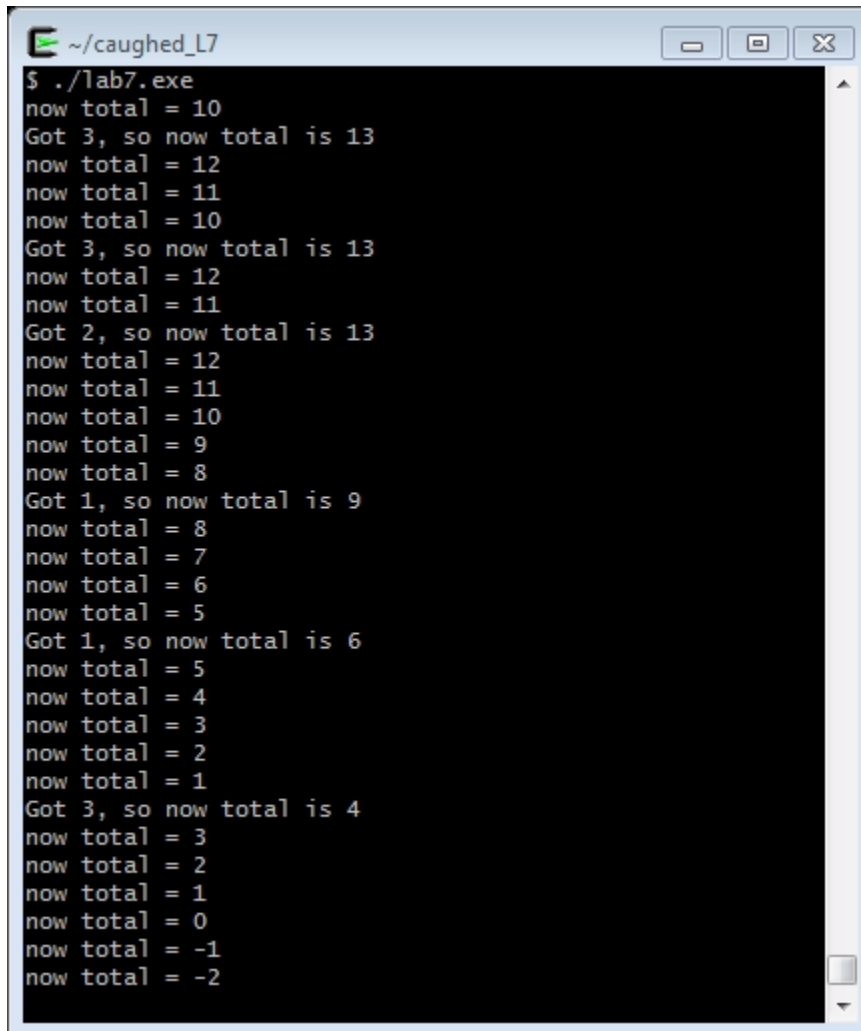
You will implement 'runAsParent()', such that it will

1. Send a random number (between 1 and 3) to the child process (via fdWrite). You will send it as a null-terminated string (e.g., "3") not an actual integer (e.g., 3)
2. Do a blocking read (via fdRead) waiting for a response from the child
 - a. If that response is less than or equal to zero, you will quit.
 - b. Otherwise, you will sleep for a random number of seconds (between 1 and 5)
3. Repeat steps 1-2 until you eventually hit step 2a

You will implement 'runAsChild()', such that it will

1. Initialize an integer "total" counter to 10
2. Configure the read file descriptor (fdRead) as non-blocking (doing appropriate error checking)
3. You will then loop, trying to read something from the input file descriptor
 - a. If the read succeeds, you will
 - i. 'sscanf' the received null-terminated string as an integer, and
 - ii. add the received value to your total, and print out the updated total, and
 - iii. send the updated total (as a null-terminated string) to the parent (via fdWrite)
 - b. If the read fails, you'll check 'errno'
 - i. If it was 'EAGAIN', call an "idle" task that will print out the current value of total, decrement the value of total, and then sleep for 1 second
 - ii. If it was not 'EAGAIN', print out an error message and quit

I should see an output that looks similar to the following (your random numbers will make it slightly different).



```
~/caughed_L7
$ ./lab7.exe
now total = 10
Got 3, so now total is 13
now total = 12
now total = 11
now total = 10
Got 3, so now total is 13
now total = 12
now total = 11
Got 2, so now total is 13
now total = 12
now total = 11
now total = 10
now total = 9
now total = 8
Got 1, so now total is 9
now total = 8
now total = 7
now total = 6
now total = 5
Got 1, so now total is 6
now total = 5
now total = 4
now total = 3
now total = 2
now total = 1
Got 3, so now total is 4
now total = 3
now total = 2
now total = 1
now total = 0
now total = -1
now total = -2
```

Transmitting Data

Information will be sent from Parent->Child and vice versa as null-terminated strings (i.e., were not going to do binary data). This means that in order to send an integer, you'll want to

```
sprintf( buf, "%d", n );
write( fdWrite, buf, strlen( buf ) + 1 );      /* send the NULL, too */
```

This means that upon successfully reading a string, you'll need to convert it to an integer

```
sscanf( buf, "%d", &n );
```

Include Files

You'll have to google stuff to figure out which include files you need use in order to get your program to compile.

Requirements

1. Create a folder called `algonquinUserID_L7` (e.g., `"myna0123_L7"`). Do all of your work in this folder, and when complete, submit the zipped folder as per the "Lab Instructions" posted on Blackboard. *DON'T USE YOUR STUDENT NUMBER. I will deduct marks if you name your folder with student number or forget to include your user ID.*
2. Follow the instructions in the "Statement of the Problem". The lecture notes on "IPC" are a good asset.

Marking

This assignment is out of 20

- 10 for coding correctness (i.e., correct results)
- 10 for clear comments and coding convention. Using pipes and forking can be confusing unless variables are appropriately named, and their use helpfully commented

You can also lose marks for incorrect submission (e.g., including unnecessary files in the zipped folder), compiler warnings, typographic errors (including in comments).

Submission

When you are done, submit your program to Blackboard. Make sure that you have the appropriate header in your source file(s), and have zipped up the appropriately name directory. Only include the source code (.c, .h) and your makefile (mandatory!). *Do not forget to zip up the directory... not just the files. I.e., when I open your zip file, I expect to find a folder called "myna0123_L7" that I can simply drag to a folder.* Do not include any other files (executables, stackdumps, vi "swap" files).

You **must** include a makefile, as part of your submission! When grading your reports I will unpack your zip file and type `'make'...` and if I don't end up with a `'lab7.exe'` to run, **I will not grade your assignment.**