CSE 2431

SP 23

Name: Jiahao Zhang (Zhang.12789)

HW 1

Due: Thursday, 1-26, at the beginning of class

PLEASE NOTE CAREFULLY: **Homework is due at the beginning of class**. I will allow a "grace" window of 15 minutes after class starts for homework to be turned in, and it will be counted as on time. Any submissions after 12:00 p.m. (noon) will be counted one day late. PLEASE do not wait till the last minute to start or finish homework, or to print it out. **The prudent (wise) approach is to finish homework by the night before it is due, *at the latest*.** "Excuses" such as "I had trouble with the printer," etc. will not be accepted to waive the late penalty. I do not like to be too strict, but please be professional; please do not ask me to waive the late penalty for a reason you would not ask your boss to accept late work!

All homework answers must be typed and printed out on paper (We do not accept on-time electronic submissions, UNLESS you have a valid reason (such as illness) for not coming to class; in such a case, email the instructor at green.25@osu.edu, and attach your HW1 file, and include a breif explanation of why you could not submit in person). Be sure to staple homework pages with a METAL STAPLE if there is more than one page, or points will be deducted (because if unstapled pages get separated, you will not get credit for the work on those pages). If you need to submit late, you can email the instructor as described above, and attach your HW1 file.

The goal of CSE 2431 homework is to give students an opportunity to understand the Unix/Linux **fork()** system call, which is used to create new processes, and also to give students experience with the stdlinux system along with the command line interpreter, as well as introducing students to forking processes.

You should not discuss the homework with other students; the homework must be the student's own individual work. You may use Piazza to post any questions which you have; if you reveal what you think the correct answer to a question is, or may be, be sure to leave your post private. Guidelines for Piazza and communication are given on the Syllabus.

On the CSE Linux System, under your home directory, create a directory cse2431 (use the mkdir command in stdlinux) for Systems II. Create a directory/folder called **hw1** using the command line interface (use the mkdir command). You will download several C source code files from Carmen, and create several executable files, as described below.

After creating your cse2431/hw1 folder on stdlinux, start a firefox web browser, by executing the following command (the $ symbol given at the beginning of each command is the Linux command line prompt; it is NOT part of the command, and should not be typed when entering the command to run):

$ firefox https://carmen.osu.edu/#

When the web browser starts, log in to Carmen, and go to Files > Homework > HW1 folder for CSE 2431.

Download the forktest1.c file to stdlinux.

Download the forktest2.c file to stdlinux.

Download the forktest3.c file to stdlinux.

Now, you can close the firefox web browser.

PLEASE NOTE: You are required to download the programs to stdlinux, run them (as processes!) there, and answer the questions based on running them in that environment. You cannot run them in your own local Linux environment, even if you use your own Linux distribution. Now, cd to your cse2431/hw1 directory. Use:

$ cd ~/cse2431/hw1

and copy the files you downloaded (they will download to your ~/Downloads directory on stdlinux) to your cse2431/hw1 directory (These commands will copy the forktest files you downloaded to your Downloads directory to your current (./) directory, but you must be sure that you used a cd command to your ~/cse2431/hw1 directory first, as directed at the beginning of this paragraph!!):

$ cp ~/Downloads/forktest1.c ./

$ cp ~/Downloads/forktest2.c ./

$ cp ~/Downloads/forktest3.c ./

IMPORTANT: When you answer the questions below, you should put your answers **in this docx file** after the **ANSWER:** for each question below, (not on stdlinux, but on the system where you have this HW1 file), and when you have all your answers, save this file, and print it out, or if you wish, export your file as a pdf, and then print it. Also be sure that you read and understand the file **Linux-Unix fork system call.pdf** on Carmen in the Files > Homework > HW1 folder.

Next, compile all three programs on stdlinux:

$ gcc forktest1.c -o forktest1

$ gcc forktest2.c -o forktest2

$ gcc forktest3.c -o forktest3

Now, run forktest1: $ forktest1

1. When the forktest1 process runs, a single child process is created (assuming no error occurs; if you get a message saying a child was not created, run the process again till you do not get this message). Give the line number of the statement (see the numbers in the comments) which is the first statement executed by the child process after it is created.
   ANSWER: statement 2

Now, run forktest2: $ forktest2

2. When the forktest2 process runs, a single child process is created (assuming no error occurs; if you get a message saying a child was not created, run the process again till you do not get this message). Both the parent process and the child process print a value for the ***num*** variable which is declared in the program code. Which of the two processes (***parent or child***) changes the value of num from the initial value, and then prints the changed value, and what is the value it prints?
   ANSWER: child changes the num value and prints 25

3. Why does the process which prints the original value of num not print the changed value (In other words, how can the process which prints the original value still access the original value after the value is changed by the other process; note that the correct answer has ***NOTHING*** to do with the difference between passing a parameter by value versus passing a parameter by reference)?
   ANSWER: because two processes don't share the same memory

Now, run forktest3: $forktest3

4. For this question, you need to count the number of processes created by the code when it runs, including the parent process (which is created by the shell when you run forktest3). Including the parent, how many processes are created?
   How many processes are created when the program forktest3 runs (Do not include in your count the process created to run the program when you execute it from the command line on stdlinux)? [**DO NOT JUST COUNT pids!** You need to **justify** this answer with your explanation for Question 5 below.]
   ANSWER: Including the parent: 8 → not including the parent: 7

F
-
T

5. Explain the parent-child relationships between the processes created when the program forktest3 executes (the process which is forked by the shell to run forktest3 as a process should be called the parent process). Give a verbal (word-based) description of the relationships. Explain, as clearly as possible (and clarity counts!) how the execution of the C code in forktest3.c results in the creation of processes which have the parent-child relationships you identify. You should use these terms to describe the relationships: a child of the parent process, of course, is called a child; for a child of a child, you can use the term grandchild, and for a child of a grandchild, you can use the term great-grandchild.
   You should also number the processes of each type, starting with the first one created, or if the order is in which the processes are created cannot be determined, the numbers do not have to correspond to the order (for children, child1, child2, etc.; for

grandchildren, granchild1, grandchild2, etc.; for great-grandchildren, great-grandchild1, great-grandchild2, etc.). Specifically, how many grandchild processes are created by child1, by child 2, etc., and how many great-grandchild processes are created by grandchild1, grandchild2, etc.? You cannot always definitely determine the order in which the processes are created, so the numbers are not meant to refer to ordering (That is, child1 is not necessarily created before child 2, etc.)
[See the sample answer below to know how to specify the relationships.]

ANSWER: [Sample answer, but *not correct for this question*]

Parent forks child1, …….
Child1 forks granchild1, …..
Child2 forks grandchild2, …..
Grandchild1 forks great-grandchild1, ….
And so on….

parent forks child 1 (in main())
parent forks child 2 (in 1st Code1())
parent forks child 3 (in 2nd Code1())
child2 forks grandchild1 (in 2nd Code1())
child1 forks grandchild2 (in 1st Code2())
child1 forks grandchild3 (in 2nd Code2())
grandchild2 forks great-grandchild1 (in 2nd Code2())

6. What is a **pid**?

ANSWER: The unique decimal number assigned to each process (process

a. How is a **pid** useful to the OS?

ANSWER: to uniquely identify an active process

b. A child process has two pids *associated with it* (Note: This **does not mean** a child process *has* two pids, but rather, it has two pids ***associated*** with it; see the getpid and getppid system calls below); what are these two pids which are associated with a child process?

ANSWER: its own pid and its parent pid

7. Look at the online manual page for **ps**; that is, run the command: **$ man ps** on stdlinux. What does the ps command do?

ANSWER: report a snapshot of current processes

a. Enter and run the command: $ **ps -af'**. What does ps -af do?

ANSWER: select all processes except both the processes not associated with a terminal and session leaders and print the full info listing

8. Look at the online manual page for the **kill** command; that is, run: $man kill on stdlinux. What does **kill** do?

ANSWER: send specified signal to the specified processes/process groups (default is to terminate it/them)

a. How might the kill command be useful to a user of the system for some process which is running for the user?

ANSWER: could manage processes so that allocate resources appropriately

b. How does the kill command relate to the **ps** command?

ANSWER: could use the ps command to see the running processes and their pids and reallocate resources by kill the corresponding processes using their pids

Directions for the questions below: Give a one or two sentence description, *in your own words*, of each of the following system calls:

9. fork:
   ANSWER: Create a child process which run concurrently with the process that call fork() call
10. getpid:
    ANSWER: return the process ID of the calling process
11. getppid:
    ANSWER: return the process ID of the parent of the calling process
12. wait
    ANSWER: block the execution of the calling process until one of its child process terminates

Type your answers to the questions above into this document, save it (if you wsih, you can export to pdf, but that is up to you), print it out and submit it in class on the due date.

Homework not submitted by the deadline on the due date but within 24 hours of the due date/time will be assessed a 25% late penalty of the grade for the whole homework assignment; homework is not accepted after that time (unless you have an extension from the instructor approved before the homework is due).