Simon Myhre

Lab 5

Report

1.
a)

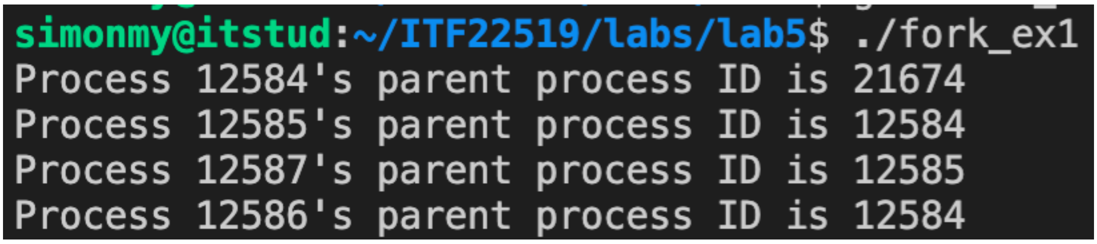
S	PID	PPID	CMD
S	21674	21600	bash
R	26668	21674	ps

b)

Parent id is 21674. Using ps -el that shows to be = bash

2.

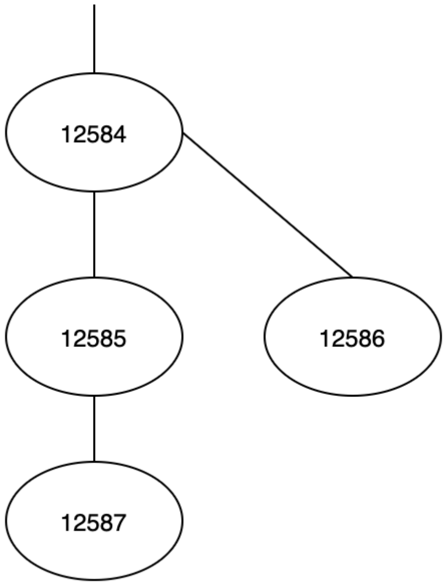
a)



```
simonmy@itstud:~/ITF22519/labs/lab5$ ./fork_ex1
Process 12584's parent process ID is 21674
Process 12585's parent process ID is 12584
Process 12587's parent process ID is 12585
Process 12586's parent process ID is 12584
```

Simon Myhre

b)



c)
First it forks 12584. Then it again forks the 12585. It has no gone down the “main” line. Then it goes back and forks the 12584 again.

By the way. Parent to 12584 is 21674. But as in the lab5.md it wasn’t included. So I didn’t too

4.
Wait() and waitpid() is used to prevent zombies. Using wait(), it will suspend a parent process until the child process is done. Then restart the parent process.
Waitpid() can do the same as wait(), but more or more specific. On success it returns 0. In wiatpid() you can change the pid. That will change what it will block (If WNOHANG it won’t block). Here are the different values:

If pid == -1, it waits for any child process. In this respect, waitpid is equivalent to wait
If pid > 0, it waits for the child whose process ID equals pid
If pid == 0, it waits for any child whose process group ID equals that of the calling process
If pid < -1, it waits for any child whose process group ID equals that absolute value of pid

https://webdocs.cs.ualberta.ca/~tony/C379/C379Labs/Lab3/wait.html

5.
It runs the ls command in the shell at /bin/ls. So the “ls” in execl is the first argument. Its possible to add even more arguments. Like -1 to the second argument. Then it would run ls -1. That explains the result. NULL is there because the execl() requires a NULL pointer.

If the execl is a success it wont return. That’s why printf() isnt performed. If execl fail then it will return, and proceed to the printf().
So, its quite simple.

Simon Myhre

6.
Arg = argument
Argc have control over the number of agruments passed
Argv is a pointer array that points to each argument passed to the program. So when we insert “op1” or “imAGenius” or whatever, it will point to that.

So that means in the for loop, the argc is used to know how many times it has to run. And the argv prints out all of its items it point to. Real simple

7.