**fork实验实验报告**

人工智能学院 202011998088 臧祝利

**实验目的**：

加深对进程概念的理解，明确进程与程序的区别，并认识并发执行的实质。

**实验要求：**

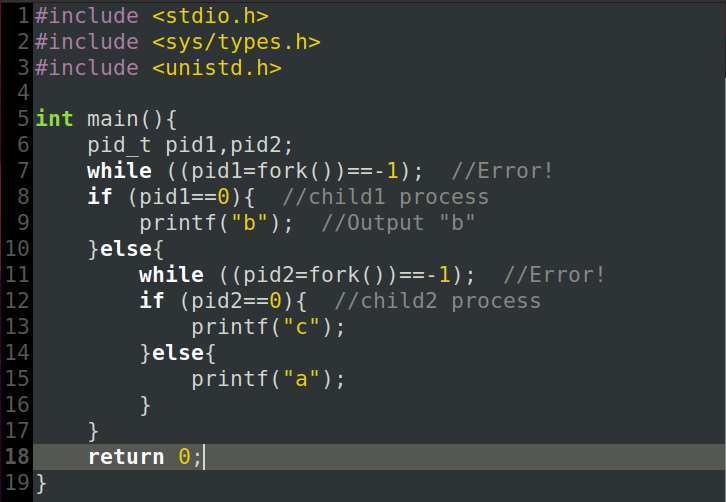
1. 编写一段程序，使用系统调用fork()创建两个子进程。当此程序运行时，在系统中有一个父进程和两个子进程活动。让每个进程在屏幕上显示一个字符，父进程显示“a”；子进程1显示“b”；子进程2显示“c”。多运行几次，观察并分析显示结果。
2. 修改程序，将每个进程输出一个字符改为每个进程输出一句话，观察分析显示结果；
3. 如果在父进程fork之前，输出一句话，这句话后面不加“\n”或加“\n”，结果有什么不同，为什么？
4. 如果在程序中使用系统调用lockf来给临界资源加锁，可以实现临界资源的互斥访问。将lockf加在输出语句前后运行试试；将一条输出语句变成多条输出语句，将lockf语句放在循环语句外部或内部试试，观察显示结果并分析原因。
5. 以上各种情况都多运行几次，观察每次运行结果是否都一致？为什么？

**思考：**

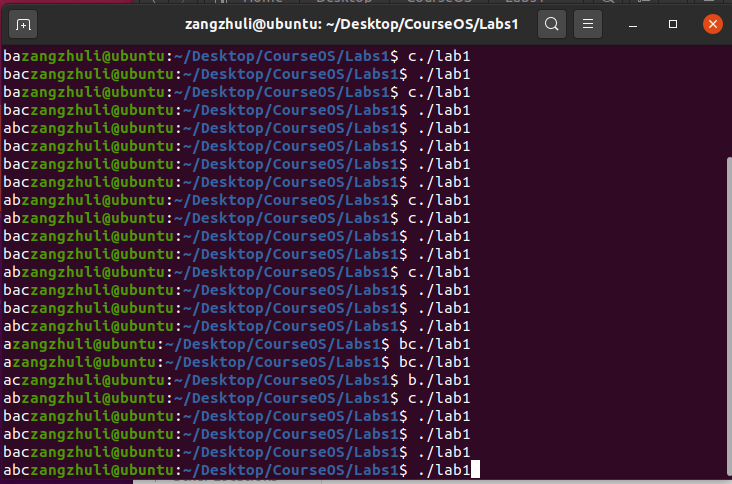
系统是如何创建进程？当父进程fork子进程后，父进程和子进程从程序什么位置开始执行？为什么？

**实验报告**

1. 编写程序**代码**如下：



**运行结果如下**；可以看到运行结果并不全部一样；（由于父进程没有等待子进程结束后再结束，因此会出现显示问题，即尚未输出完成就已经显示命令行的开头，但指令可以继续输入，没有产生其他影响）

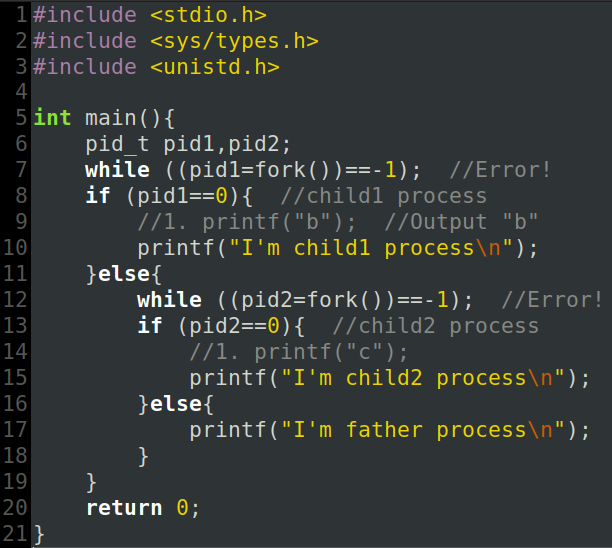


**分析：**执行fork()函数，可能返回三个结果，如果是负数说明出现错误，0说明创建子进程，正数返回的是新子进程的ID；当pid1=fork()为负数的时候，持续循环，至出现0，说明创建了子进程pid1，输出“b”，再继续向下执行pid2=fork()，创建子进程pid2，然后输出“c”，然后运行父进程，输出“a”；可以发现运行的结果并不是完全相同的。

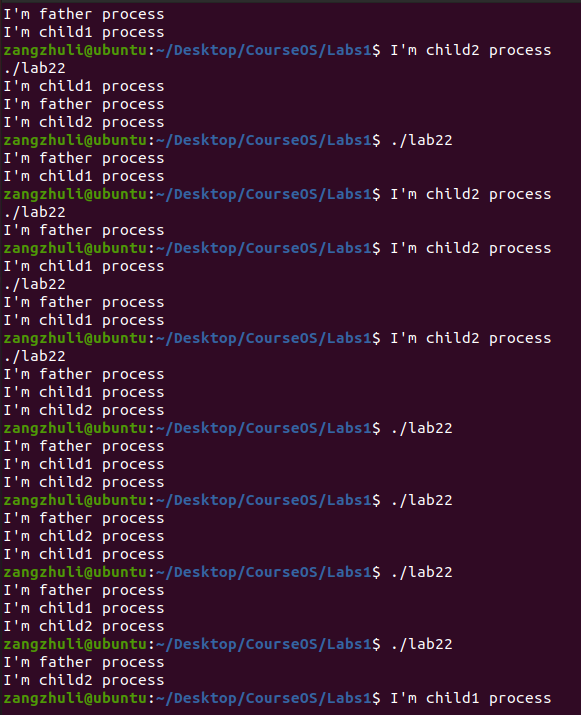
**原因：**

fork () 创建子进程后，父进程和子进程执行的先后顺序并不一定，这取决于系统是如何调度的。

2.程序**代码**如下，将输出字母更改成了输出一句话；

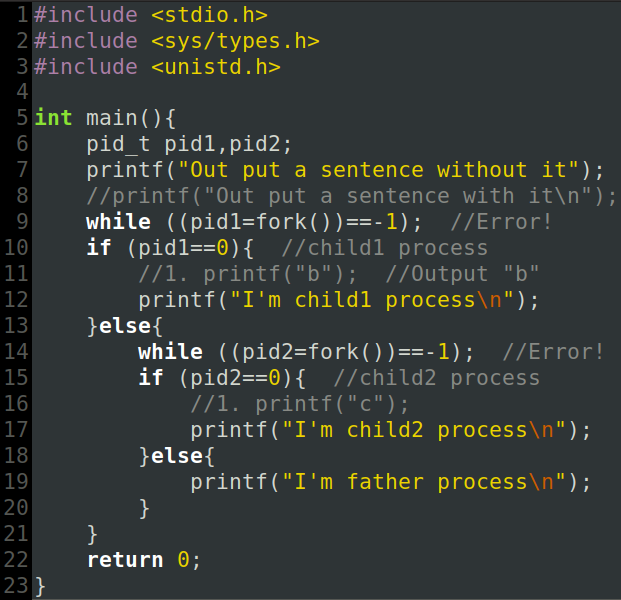


**结果**如图，和1一样，输出不尽相同；

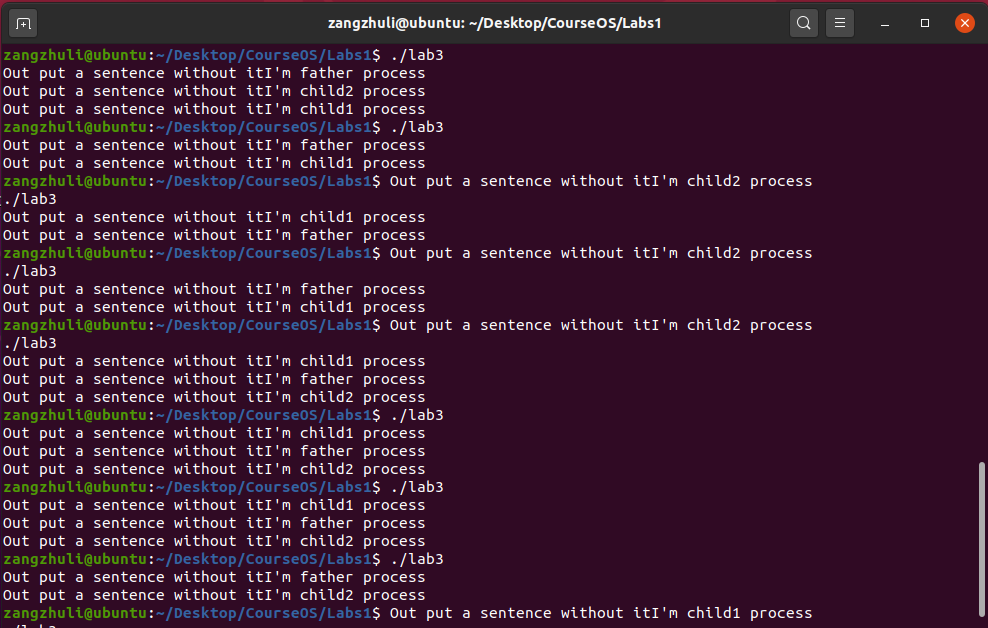


**分析：**代码将输出一个字符改为输出一句话，当遇到对应子进程或父进程时输出”child1 process”,”child2 process”和”father process”用来区分，输出顺序问题同1；

3.程序**代码**如下：

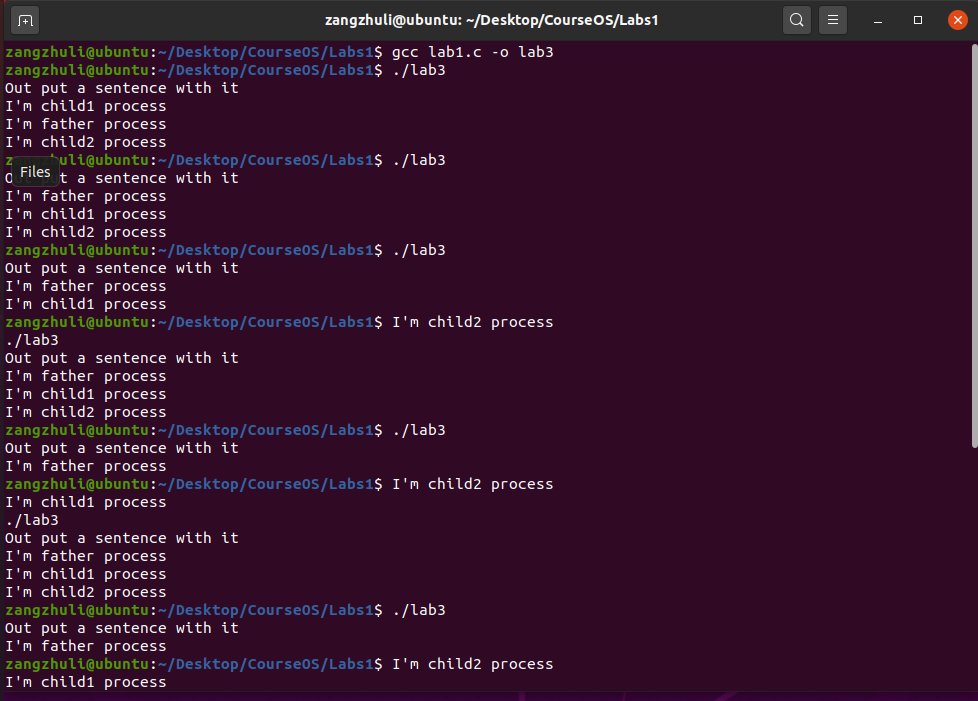


**运行结果①**，不带”\n”，



发现每一句话输出前都会输出新加入的那句话；

**运行结果②**，输出的内容带”\n”



根据结果，发现每一次运行此程序只会输出一次新加入的那句话；

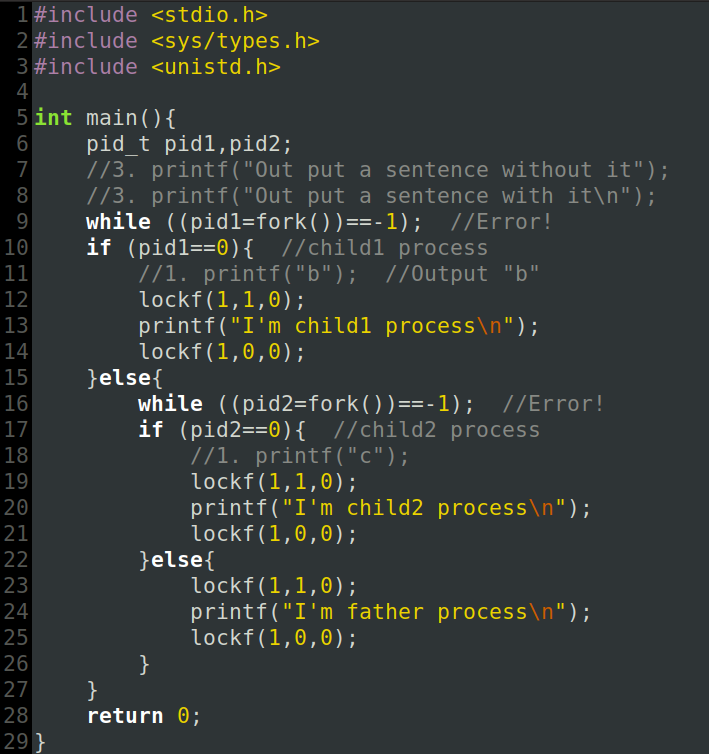
**原因：**

printf的缓冲是行缓冲，意思就是当行缓冲里被填满的时候就把缓冲里的内容打印输出，所以当没有'\n'的时候，缓冲没有满，子进程会继承父进程缓冲区的内容，因此也会输出相同的内容；

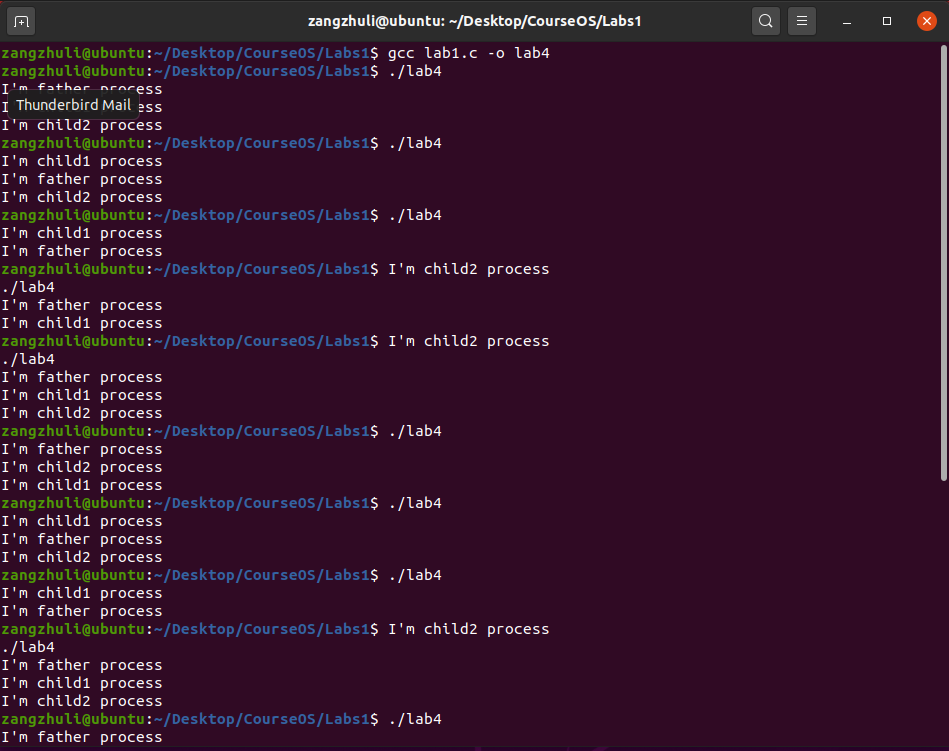
当有'\n'的时候，就相当于缓冲满了，会打印输出，并清空缓冲区，因此子进程的缓冲区中没有这句话，就只会输出一次。

1. ①将lockf加在输出语句前后运行：

**代码：**



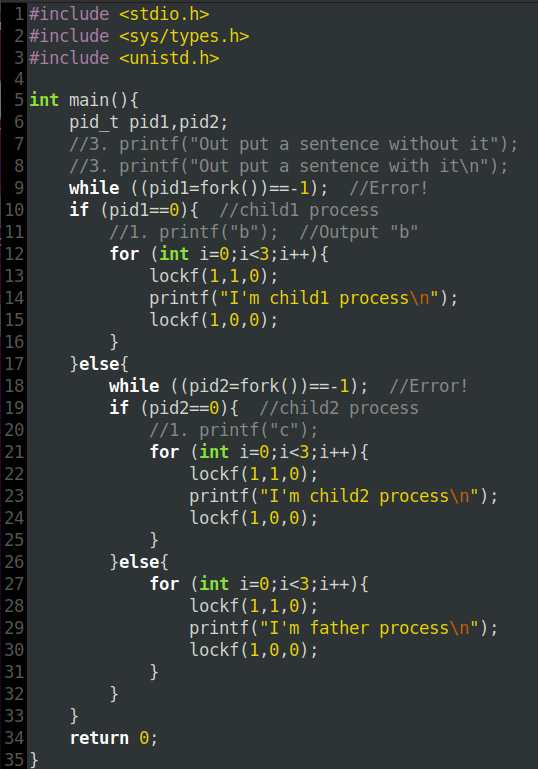
**结果：**



和lockf前没有任何区别，输出结果都比较乱序，没有规律可言；lockf可以实现资源的互斥访问，一个进程在输出时另一个进程无法输出，但是只有一句话，因此看不出有什么特点；

②加到循环内部：

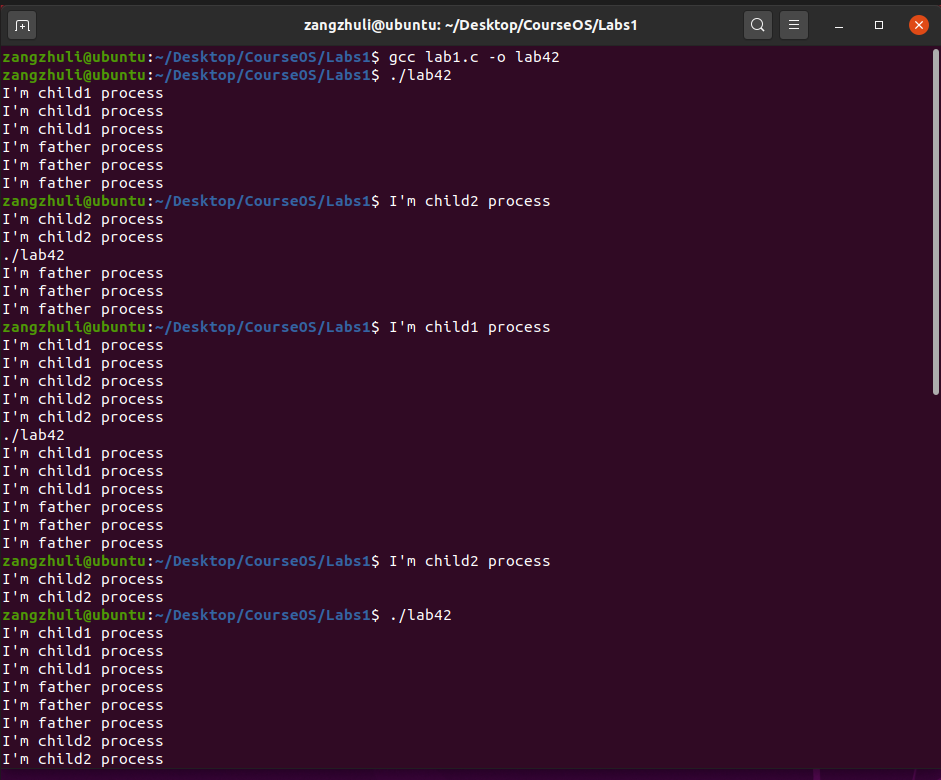
**代码**如下，让每个句子循环3次：



**结果**如图所示，每个句子都会循环输出完后才输出别的句子；

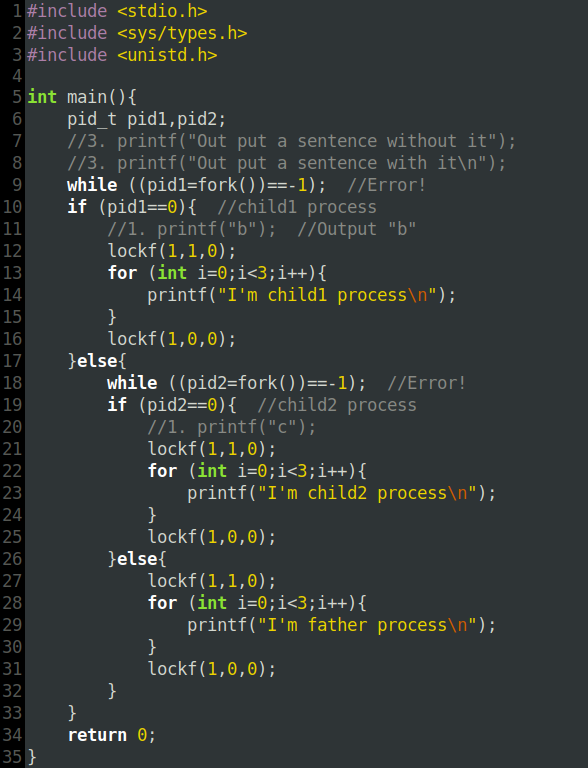
**原因：**

父进程或子进程被加锁后，能够防止二者对输出资源进行竞争，因此每一次只会有一个进程先将循环输出完再到下一个进程输出；放在循环内部，加锁后输出，释放锁，由于循环又会加锁，因此会一直执行一个进程的内容；



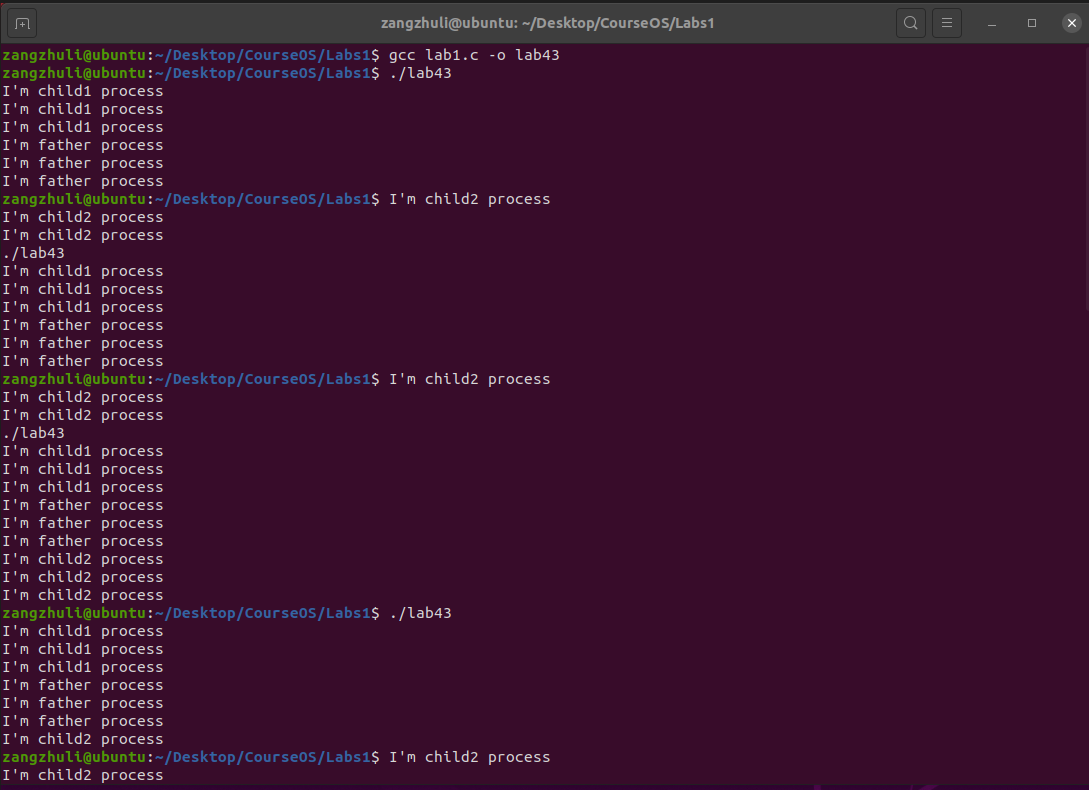
③放在循环外部：

**代码：**



**结果**如下，发现放在循环外与循环内没有区别；

**原因：**循环外部加锁，整个循环过程中输出资源不能被其他进程访问，一直到循环结束，因此会一直输出同一句话。



1. 以上各种情况都多运行几次，观察每次运行结果是否都一致？为什么？

每次运行结果都不一致；

原因：父子进程的运行顺序并不固定，由系统的调度算法决定。

**思考：**

系统是如何创建进程？当父进程fork子进程后，父进程和子进程从程序什么位置开始执行？为什么？

系统首先会创建父进程，然后创建子进程，子进程会继承父进程的资源和环境。

子进程是从fork后面那个指令开始执行的；

**原因：**以实验中为例，在fork前输出额外一句话，并带上’\n’时，只输出了一次这句话，如果从头开始重新运行的话，肯定会再次输出；从实际思路出发，如果每次都重新执行，那么每次都会fork出一个子进程，会出现无穷多的子进程，这实际上是不可能的，因此是从fork后的指令开始运行。