实验二 Linux进程控制1

**学号： 20110031020 姓名： 许健 年级： 20级**

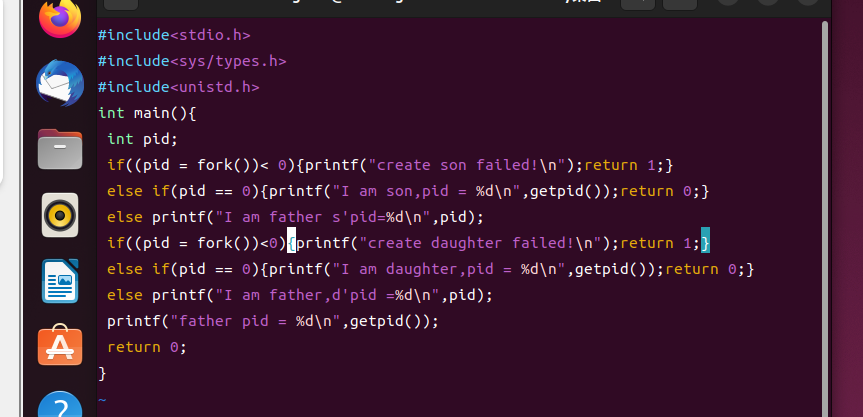
1. **使用fork系统调用创建多个子进程**

**1.具体要求与步骤**

1. 编写一C/C++语言程序（程序名为fork.c/fork.cpp），使用系统调用fork()创建两个子进程。当程序运行时，系统中有一个父进程和两个子进程在并发执行。父亲进程执行时屏幕显示“I am father”，儿子进程执行时屏幕显示“I am son”，女儿进程执行时屏幕显示“I am daughter”。
2. 多次连续反复运行这个程序，观察屏幕显示结果的顺序，直至出现不一样的情况为止。记下这种情况，试简单分析其原因。
3. 可以使用实验报告模板中所推荐的代码实现，但是要求为代码添加注释，对代码关键逻辑步骤进行解释。在代码头部添加如代码1所示式样的头部版权声明。使用星号、井号、等号、破折号等各类符号对版权声明添加边框，并拼出19os的式样。

**2.截图：**

1）代码：



首先在父进程内创建一个子进程：

对于父进程来说，会打印子进程的pid；

对于子进程来说，会打印自身的pid,并结束子进程；

然后在父进程内创建一个女进程:

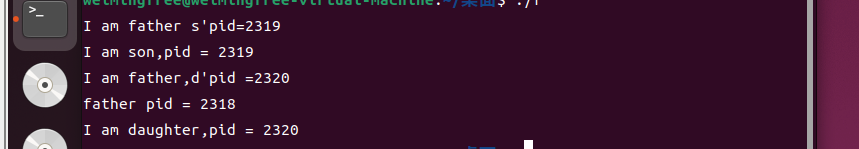
对于父进程来说，会打印女进程的pid；

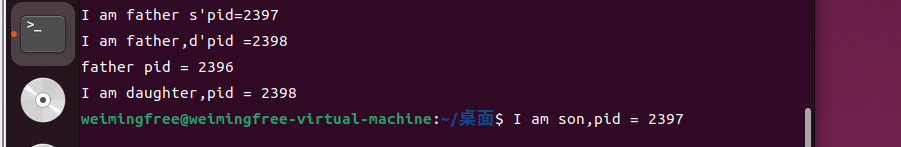
对于女进程来说，会打印自身的pid,并结束女进程；

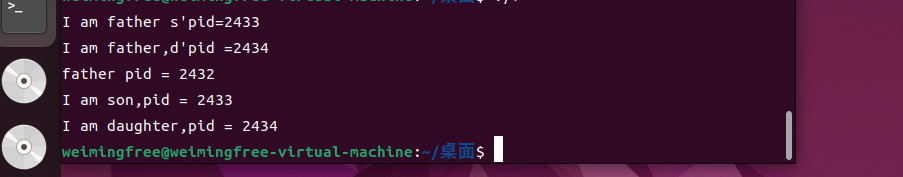
最后会打印父进程自身的pid，并结束父进程。

2)验证:

多次运行结果:







打印顺序不相同的原因：没有对三个进程对打印权执掌的顺序进行规定，导致进程之间互相争夺，通过打印结果来看，父进程在打印权争夺中竞争力更强，而子进程和女进程竞争力差不多。

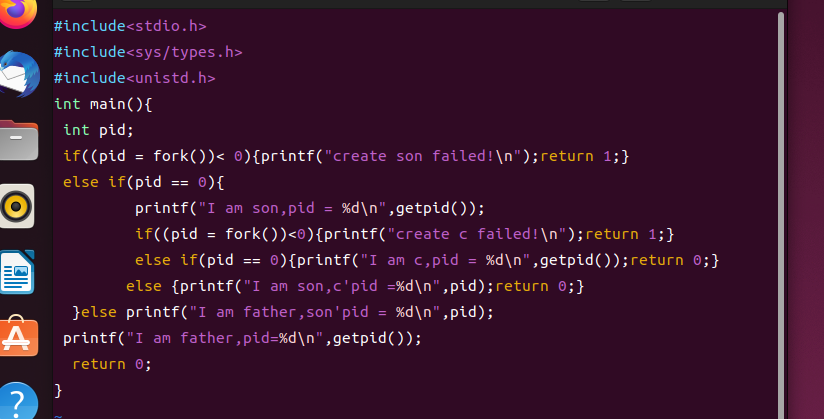
1. **使用fork系统调用创建多级子进程**

**1.具体要求与步骤**

1. 编写一C/C++语言程序（程序名为fork.c），使用系统调用fork( )创建一个子进程，然后在子进程中再创建子子进程。当程序运行时，系统中有一个父进程、一个子进程和一个孙子进程在并发执行。父亲进程执行时屏幕显示“I am father”，儿子进程执行时屏幕显示“I am son”，孙子进程执行时屏幕显示“c”。
2. 多次连续反复运行这个程序，观察屏幕显示结果的顺序，直至出现不一样的情况为止。记下这种情况，试简单分析其原因。
3. 可以使用实验报告模板中所推荐的代码实现，但是要求为代码添加注释，对代码关键逻辑步骤进行解释。在代码头部添加如代码1所示式样的头部版权声明。使用星号、井号、等号、破折号等各类符号对版权声明添加边框，并拼出19os的式样。

**2.截图**

1)代码:



首先在父进程内创建一个子进程：

对于父进程来说，会打印子进程的pid；

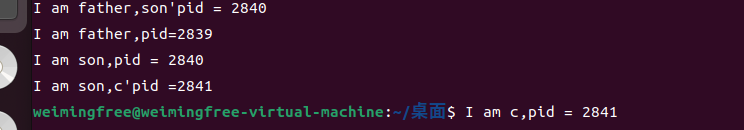
对于子进程来说，会打印自身的pid,然后在子进程内创建一个孙进程:

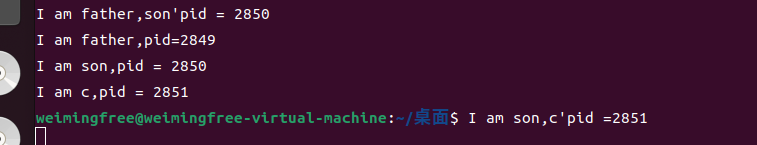
对于子进程来说，会打印孙进程的pid，并结束子进程；

对于孙进程来说，会打印自身的pid，并结束孙进程。

最后会打印父进程自身的pid，并结束父进程。

2)打印结果:







在大多数情况下是第一种结果，说明父进程的抢夺打印权的能力>子进程>孙进程

但有时候孙进程也会先于子进程抢到打印权,子进程也会先于父进程抢到打印权。

我们还能注意到，对于一个进程来说，其内部打印相对顺序是不变的；比如，对于父进程来说，”I am father,son’pid =”永远比”I am father,pid =”要先输出；对于子进程来说，“I am son,pid =”永远比“I am son,c’pid = ”要先输出，这与程序书写的顺序相同。