**201311211918\_汪大洋**

**实验目的**：进一步认识并发执行的实质；分析进程争用资源的现象

**实验要求：**

    1. 修改实验二的程序，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；

答：将每个进程中输出的字符改为一句话的时候，实验的结果并没有太大的差别，这几句话任然会间断的出现，而且这些话中间不会断。于是设想当一句话很长的时候可能会断掉。

#include<stdio.h>

int main()

{

int i,p1,p2;

printf("Ola!\n");

p1=fork();

if(!p1)

{

for(i=0;i<5;i++)

{

printf("p1 is working%d\n",i);

sleep(1);

}

}

else {

p2=fork();

if (!p2) {

printf("p2 is working\n");

else {

printf("parent is working\n");

}

}

//return 0;

}

2. 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？

答：如何在fork之前加上一句话，这句话就会为父进程与两个子进程所共有，不加“\n”的话这句话会被输出三遍。如果加“\n”，这句话只会在一开始输出一遍。

因为子进程继承了父进程所有的资源，包括缓冲区里的资源。而加了“\n”之后，缓冲区被flash，就只会输一遍了。

3. 如果在程序中使用系统调用lockf来给每一个进程加锁，可以实现进程之间的互斥。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， 观察分析显示结果。

答：将lock(1,1,0)语句放在输出语句前，将lock(1,0,0)语句放在输出语句后会对该输出语句加锁，在该锁内输出的内容不会被其他进程输出的内容插入。（调用wait语句来帮助实验）

将lockf语句放在循环外部的时候，循环内部的内容输出的时候不会被中断，lockf语句放在循环内部的时候，在lockf内部的输出内容不会被其他进程插入。

注：当使用lockf语句时，必须对所有的lockf语句使用lockf语句，这类似一种规范的协议，当有输出语句未使用lockf语句时，它们可能插入到使用了lockf的输出结果中。

#include<stdio.h>

int main()

{

int i,p1,p2;

printf("Ola!\n");

p1=fork();

if(!p1)

{

lockf(1,1,0);

for(i=0;i<5;i++)

{

printf("p1 is working%d\n",i);

sleep(1);

}

lockf(1,0,0);

}

else {

p2=fork();

if (!p2) {

lockf(1,1,0);

printf("p2 is working\n");

lockf(1,0,0);}

else {

lockf(1,1,0);

printf("parent is working\n");

lockf(1,0,0); }

}

//return 0;

}

4.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？

答：将程序多运行几次会发现，在大部分的情况下，运行结果是一样的，但是也有输出荀淑不一样的情况。因为进程在多任务的操作系统中的执行顺序是未知的，不可再现的。

**思考与体会**

当父进程fork子进程后，父进程和子进程如何执行程序的？

答：当父进程fork之后，父进程会得到一个返回值（子进程的pid），子进程得到返回值0，父进程和子进程同时向下执行，一般有如下代码：

P=fork（）；

if(!p) {子进程的执行内容}

else {父进程的执行内容}

**实验作业**

     上交一份实验报告。报告包括：程序代码、运行结果、实验分析。

     实验报告命名方式：学号\_姓名.doc

    实验报告通过Email上交。Email地址：xiaorong@bnu.edu.cn