实验二 进程创建及进程间通信

一、 实验目的

1. 掌握Linux进程的创建方法，加深对进程概念的理解，明确进程和程序的区别。
2. 认识进程并发执行的实质。
3. 学习控制进程同步的方法。
4. 分析进程竞争资源的现象，学习解决进程互斥的方法。
5. 了解管道通信的特点，掌握管道通信的使用方法。

二、实验内容

1. 运行Linux进程的创建程序，观察运行结果。
2. 利用`fork`函数，编写程序。
3. 用`fork( )`创建一个进程，再调用`exec( )`用新的程序替换该子进程的内容。利用`wait( )`来控制进程执行顺序。
4. 用`lockf( )`来给每一个进程加锁，以实现进程之间的互斥。
5. 用`pipe( )`来实现进程的管道通信。

三、实验步骤

1.进程创建程序示例：

```
 #include<stdio.h>
    main(){
    int  p1;
    while((p1=fork())= = -1);
      if(p1= =0){/*在子进程中*/
         printf(“This is a child process.”);
      } else{ /*在父进程中*/
            printf(“This is a parent process.”);
      }
    return 0;
}
```

2.进程的创建。

编写一段C/C++程序，使用系统功能调用`fork( )`创建两个子进程。

要求：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。让每一个进程在屏幕上显示一个字符：父进程显示字符`a`，子进程分别显示字符`b`和`c`。试观察记录屏幕上的显示结果，并分析原因。
提示：实验结果需要运行多次找出不同结果并分析原因。

3.运行以下程序，并分析`switch`语句中各个`case`所做的事和产生原因。

`wait( )`给我们提供了一种实现进程同步的简单方法，它是如何实现进程同步的?

```
#include<stdio.h>
#include<unistd.h>
int main(){
    int pid;
    pid = fork();/*创建子进程*/
    switch(pid){
        case -1:
            printf("Error in fork()\n");
            exit(1);
        case 0:
            execl("/bin/ls","ls","-l","-color",NULL);
            printf("execl fail!\n");
            exit(1);
        default:
            wait(NULL);
            printf("is completed!\n");
```

```
            exit(0);
        }
    }
```

4.分析以下程序的输出结果。

可以使用`cat to_be_locked.txt`查看输出结果。多次验证看是否有不同结果，为什么?

```
#include <stdio.h>
#include <unistd.h>

int main(){
    int p1,p2,i;
    int *fp;
    fp = fopen("to_be_locked.txt","w+");
    if(fp==NULL){
        printf("Fail to create File\n");
        exit(-1);
    }
    while((p1 = fork())==-1){ /*创建子进程p1*/
        if(p1==0){
            lockf(*fp,1,0); /*加锁*/
            for(i=0;i<10;i++)
                fprintf(fp,"daughter %d\n",i);
            lockf(*fp,0,0); /*解锁*/
        }else{
            while((p2 = fork())==-1){ /*创建子进程p2*/
                if(p2==0){
                    lockf(*fp,1,0);/*加锁*/
                    for(int i=0;i<10;i++)
                        fprintf(fp,"son %d\n",i);
                    lockf(*fp,0,0);/*解锁*/
                }else{
                    wait(NULL);
                    lockf(*fp,1,0);/*加锁*/
                    for(i = 0;i<10;i++)
                        fprintf(fp,"parent %d\n",i);
                    lockf(*fp,0,0);/*解锁*/
                } // end if p2 == 0
            }; // end p2
        } // end if p1 == 0
    }// end p1
    fclose(fp);
}
```

5.编写程序：实现进程的管道通信.

用系统调用`pipe( )`建立一管道，二个子进程P1和P2分别向管道各写一句话：
Child 1 is sending a message!
Child 2 is sending a message!
父进程从管道中读出二个来自子进程的信息并显示。
要求:先接收P1，后P2。

操作系统实验仓库

github仓库地址
gitee码云仓库地址

详情请查看仓库中`readme.md`.实验手册与实验报告模板已经更新至ex2。