**实验二 进程的控制**

# 姓 名 学 号 成绩

实验时间 指导教师(签名)

**（诚信声明：本实验报告内容，均由本人亲自上机完成。 签名：\_\_\_\_\_\_\_\_\_\_\_\_\_）**

一．实验目的

1.熟悉和理解进程和进程树的概念，掌握有关进程的管理机制

2.通过进程的创建、撤销和运行加深对进程并发执行的理解

3.明确进程与程序、并行与串行执行的区别

4.掌握用C 程序实现进程控制的方法

二．实验工具与设备

已安装 Linux 操作系统的计算机。

1. 实验内容

## 1. 了解系统调用 fork( ), exec 系列函数, exit( ), waitpid( )的功能和实现过程

## 2．程序设计

* 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’。观察屏幕上的显示结果，并分析原因。

代码：

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

int main() {

pid\_t child\_pid1, child\_pid2;

child\_pid1 = fork(); // 创建第一个子进程

if (child\_pid1 == 0) {

// 子进程1

printf("Child Process 1 'b'\n");

} else {

child\_pid2 = fork(); // 创建第二个子进程

if (child\_pid2 == 0) {

// 子进程2

printf("Child Process 2 'c'\n");

} else {

// 父进程

printf("Parent Process 'a'\n");

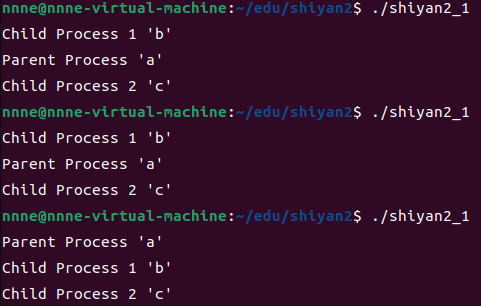
}

}

return 0;

}

实验结果：



原因：

使用fork()创建进程时，fork会进行两次返回，随机返回到父进程或者子进程，返回到父进程时，返回值是子进程的PID，返回到子进程时，返回值为0。以该返回值是否大于零，判断返回到了子进程还是父进程，并执行打印语句。由于fork()会随机返回，所以abc的打印顺序也是随机的。

* 1. 编写一段程序，使用系统调用 fork( )创建一个子进程。子进程通过系统调用 exec 系列函数调用命令 ls，调用 exit( )结束。而父进程则调用 waitpid( )等待子进程结束，并在子进程结束后显示子进程的标识符，然后正常结束。

# 代码：

# #include <stdio.h>

# #include <stdlib.h>

# #include <unistd.h>

# #include <sys/types.h>

# #include <sys/wait.h>

# int main() {

# pid\_t child\_pid;

# child\_pid = fork();

# if (child\_pid == 0) {

# printf("Child Process (PID: %d) 'ls'\n", getpid());

# execl("/bin/ls", "ls", NULL);

# perror("Exec failed");

# exit(1);

# } else {

# int status;

# waitpid(child\_pid, &status, 0);

# if (WIFEXITED(status)) {

# printf("Child Process (PID: %d) end\n", child\_pid);

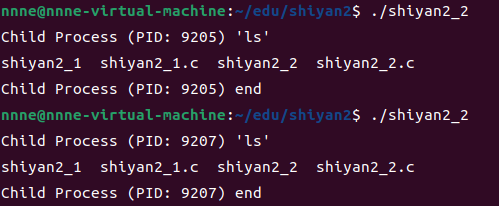
# }

# }

# return 0;

# }

# 实验结果：



# 思考题

## 怎样用 C 程序实现进程的控制？当首次调用新创建进程时，其入口在哪里？

## 使用fork(),wait(),waitpid(),exec族函数,exit()等函数实现进程的控制。

首次调用新创建进程时，入口在fork()函数的下一行代码。

1. **系统调用 fork( )是如何创建进程的？系统调用 exit( )是如何终止一个进程的？**

调用 fork 时，系统将创建一个与当前进程相同的新进程，其与原有进程具有相同的数据、连接 关系和在程序同一处执行的连续性。原进程称为父进程，新生进程称为子进程。子进程是父进程的 一个拷贝，子进程获得同父进程相同的数据，但是同父进程使用不同的数据段和堆栈段。子进程被 创建以后，处于可运行状态，与父进程以及系统中的其他进程平等地参与系统调度。 fork 调用将执行两次返回，即分别从父进程和子进程分别返回，即子进程一旦创建就绪，就与 父进程一样被平等地调度执行。因此，从 fork 返回以后，不能确切知道执行哪一个进程。从父进程 返回时，返回值（>0）为子进程的进程标识号 PID；而从子进程返回时，返回值为 0，并且返回都 将执行 fork 之后的语句。调用出错时返回值为-1，并将 errno 置为相应值。

exit()会释放进程所占用的资源，包括内存，文件描述符等，并将终止状态返回给父进程，这个状态值可以通过wait()函数来获取，它反映了进程的结束状态

1. **系统调用 exec 系列函数是如何更换进程的可执行代码的？**

exec()函数会将一个新的可执行文件读取到内存中，新程序加载完成后，exec()

会清除当前进程的内存空间，以便为新程序的执行腾出空间。exec()将新程序的指令和数据加载到当前进程的内存空间，然后将控制权转交给新程序，从其入口开始执行。这意味着原进程的执行流程已经被替换为新程序的执行流程。