**《操作系统实验》课程题目（2020-2021学年春季学期）**

**评分标准：本课程共布置5道实验题目，每道20分。根据每道题是否达到实验要求、输入是否灵活、输出是否美观酌情给分。**

**1 进程管理（20分）**

**上机准备：**

 熟悉 Cygwin 环境

 编译源程序使用 gcc 源程序名 –o 目标文件名（缺省为

a.out）。

 学习 Linux（Unix）的常用命令。

 学习 vi 编辑器的使用。

 C 语言语法

**1.1 实验目的**

加深对于进程并发执行概念的理解。实践并发进程的创建和控制方法。观察和体验进程的动态特性。进一步理解进程生命期期间创建、变换、撤销状态变换的过程。掌握进程控制的方法，了解父子进程间的控制和协作关系。练习Linux系统中进程创建与控制有关的系统调用的编程和调试技术。

**1.2 实验说明**

1）与进程创建、执行有关的系统调用说明

进程可以通过系统调用fork()创建子进程并和其子进程并发执行.子进程初始的执行映像是父进程的一个复本.子进程可以通过exec()系统调用族装入一个新的执行程序。父进程可以使用wait()或waitpid()系统调用等待子进程的结束并负责收集和清理子进程的退出状态。

fork()系统调用语法:

#include <unistd.h>

pid\_t fork(void);

fork成功创建子进程后将返回子进程的进程号,不成功会返回-1.

exec 系统调用有一组6个函数,其中示例实验中引用了 execve 系统调用语法:

#include <unistd.h>

int execve(const char \*path, const char \*argv[], const char \* envp[]);

path 要装入的新的执行文件的绝对路径名字符串.

argv[] 要传递给新执行程序的完整的命令参数列表(可以为空).

envp[] 要传递给新执行程序的完整的环境变量参数列表(可以为空).

Exec执行成功后将用一个新的程序代替原进程，但进程号不变，它绝不会再返回到调用进程了。如果exec调用失败，它会返回-1。

getpid()---获取进程的pid

* 每个进程都执行自己独立的程序，打印自己的pid，每个父进程打印其子进程的pid;

**父**

**子1**

**子2**

* 每个进程都执行自己独立的程序，打印自己的pid，父进程打印其子进程的pid;

**父**

**子1**

**子2**

* 编写一个命令处理程序，能处理max(m,n), min(m,n)和 average(m,n,l)这几个命令。（使用exec函数族）