 ****

**计算机科学与技术学院**

**《操作系统》实验报告**

（ 2019 / 2020 学年 第 二 学期）

**学生姓名： XXX**

**学生专业： 计算机科学与技术**

**学生班级： 计算机172002**

**学生学号： 201720XXXXX**

2020 年 5 月 31 日

# 实验一 windows进程的创建

实验学时：2

实验类型：验证

实验要求：必修

**一、实验目的**

本课题实习的目的是，加深对windows进程概念及进程管理各部分内容的理解；熟悉windows进程管理API的使用。

**二、实验要求**

1. 将源程序编译、链接后形成master.exe和slave.exe文件。
2. 读懂程序，并加关键代码的注释。
3. 在命令行方式下输入……> master slave回车，将在master进程中创建slave进程，观察程序运行的结果。
4. 自己设计一个小程序，完成在master进程中启动该程序的操作。
5. 撰写实验报告。

**三、实验内容**

1.下面程序是两个简单的控制台应用程序，第一个程序（ MASTER ）运行第二个程序（ SLAVE ）, 并进入睡眠。SLAVE 程序从命令行读取MASTER程序的进程 ID(PID), 并等待 MASTER 程序终止之后，SLAVE做了一些事情之后，也结束。

**2.** 自己设计一个小程序，完成在master进程中启动该程序的操作。

**四、源程序清单：**

1.

#include "m.h"

#define \_CRT\_SECURE\_NO\_DEPRECATE

#include<iostream.h>

#include<windows.h>

#include<stdio.h>

#include<string.h>

void main(int argc, char\* argv[])

{

char cmd[128];

if (argc != 1)

strcpy(cmd, argv[1]);

else

strcpy(cmd, "slave.exe");

int pid = GetCurrentProcessId();//获取当前进程一个唯一的标识符

cout << "process ID" << pid << endl;

cout << "Master 准备启动：" << cmd << "\n";

sprintf(cmd+strlen(cmd), " %d", pid);

cout.flush();

STARTUPINFO info;

memset(&info, 0, sizeof(info));//初始化变量

info.cb = sizeof(info);

PROCESS\_INFORMATION pinfo;//有关新进程及其主线程的信息

if (!CreateProcess(NULL, cmd, NULL, NULL, FALSE, NORMAL\_PRIORITY\_CLASS, NULL, NULL, &info, &pinfo))//没有找到进程

{

cout << "Master:从进程" << cmd << "没有找到"<<endl;

cout << "Master:重新输入进程名\n";

}

cout << "Master:睡觉..........." << endl;

cout.flush();//刷新缓冲

Sleep(1000);

for (int i = 1; i < 5; i++)

{

cout << "睡醒了" << endl;

}//循环输出

cout << "Master:刷牙，洗脸，吃饭\n";

cout << "Master:结束！！！\n";

exit(0);//正常运行程序并退出程序

CloseHandle(pinfo.hProcess);//关闭了新进程句柄

CloseHandle(pinfo.hThread);//关闭了主线程句柄

}

2.

#include "slave.h"

#include<iostream>

#include<windows.h>

#include<stdio.h>

using namespace std;

void main(int argc, char\* argv[])

{

if (argc != 2)//保证顺序执行

{

cout << "Slave：请重新运行 MASTER.EXE" << endl;

exit(1);

}

int pid = atoi(argv[1]);//获取当前标识符

HANDLE process = OpenProcess(PROCESS\_QUERY\_INFORMATION | SYNCHRONIZE, FALSE, pid);

if (!process)

cout << "Slave:Error opening process" << endl;

cout << "Slave:我要等Master起床" << endl;

cout.flush();

if (WaitForSingleObject(process, INFINITE) == WAIT\_OBJECT\_0)//有信号时执行

cout << "Slave:master 结束了，该我了" << endl;

else

cout << "Slave:出什么错了？" << endl;

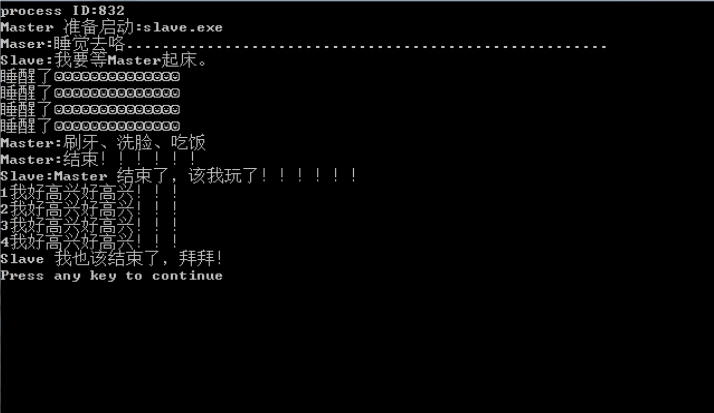
cout << "Slave:刷牙，洗脸，吃饭\n";

cout << "我也结束了" << endl;

exit(0);

}

1. **测试结果**



1. **实验总结**

本实验中有master和slave两个程序。Mater在单独执行后会先出现一部分试验运行结果，等待一段时间后再执行slave，slave却不能单独执行，因为它的执行要等master执行结束后再执行：当我们以命令行的方式在master进程中创建slave进程执行时会发现执行结果和与单独执行master是一致的。通过这次实验，我对Windows系统进程的创建有了更深入的了解，不再是原来那种非常的肤浅的理解了，这次实验也对我掌握课上学到的知识有很大的促进作用。