**操作系统课程设计实验报告**

实验名称： 进程控制

姓名/学号： 刘鑫/1120181208

1. **实验目的**

设计并实现Unix的“time”命令。“mytime”命令通过命令行参数接受要运行的程序，创建一个独立的进程来运行该程序，并记录程序运行的时间。

1. **实验内容**
2. 在Windows下实现
3. 使用CreateProcess()来创建进程
4. 使用WaitForSingleObject()在“mytime”命令和新创建的进程之间同步
5. 调用GetSystemTime()来获取子进程运行时间
6. 将获得的时间格式化输出
7. 在Linux下实现
8. 使用fork()来创建子进程运行程序
9. 使用execv()指定子进程执行的程序
10. 父进程使用wait()等待新创建的子进程结束
11. 调用gettimeofday()来获取子进程运行时间
12. 将获得的时间格式化输出
13. **实验环境**

本实验基于Windows系统和Linux虚拟机完成，具体环境配置如下

**Windows：**

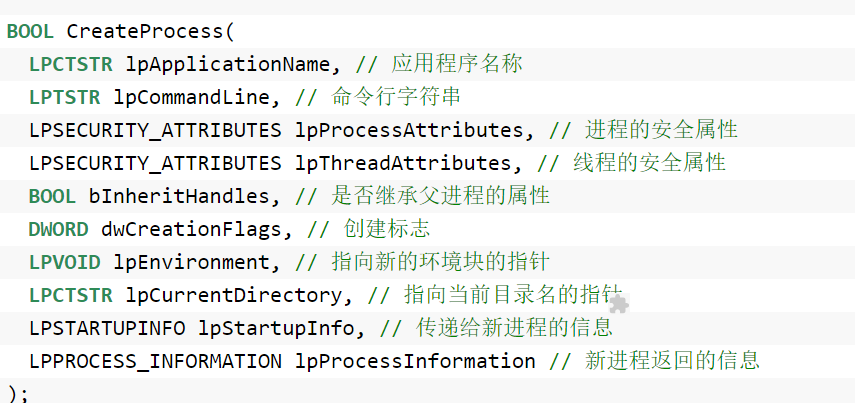
1. Windows10
2. 运行内存8GB
3. 处理器：1.8GHz Intel Core i5

**虚拟机环境：**

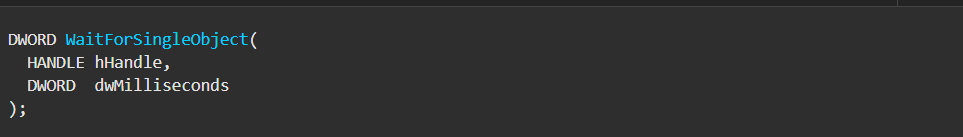
1. 虚拟机软件：VMware workstation 15
2. 虚拟机操作系统：ubuntu 20.04
3. 虚拟机内存：4GB
4. 虚拟机硬盘容量：60GB
5. **程序设计与实现**

**在Windows下的实现：**

1. 主函数调用CreateProcess()

该函数的功能是创建进程，具体参数描述如下

1. 调用WaitForSingleObject()在“mytime”命令和新创建的进程之间同步

WaitForSingleObject()作用是使父进程等待子进程，具体描述如下：

hHandle

对象的句柄

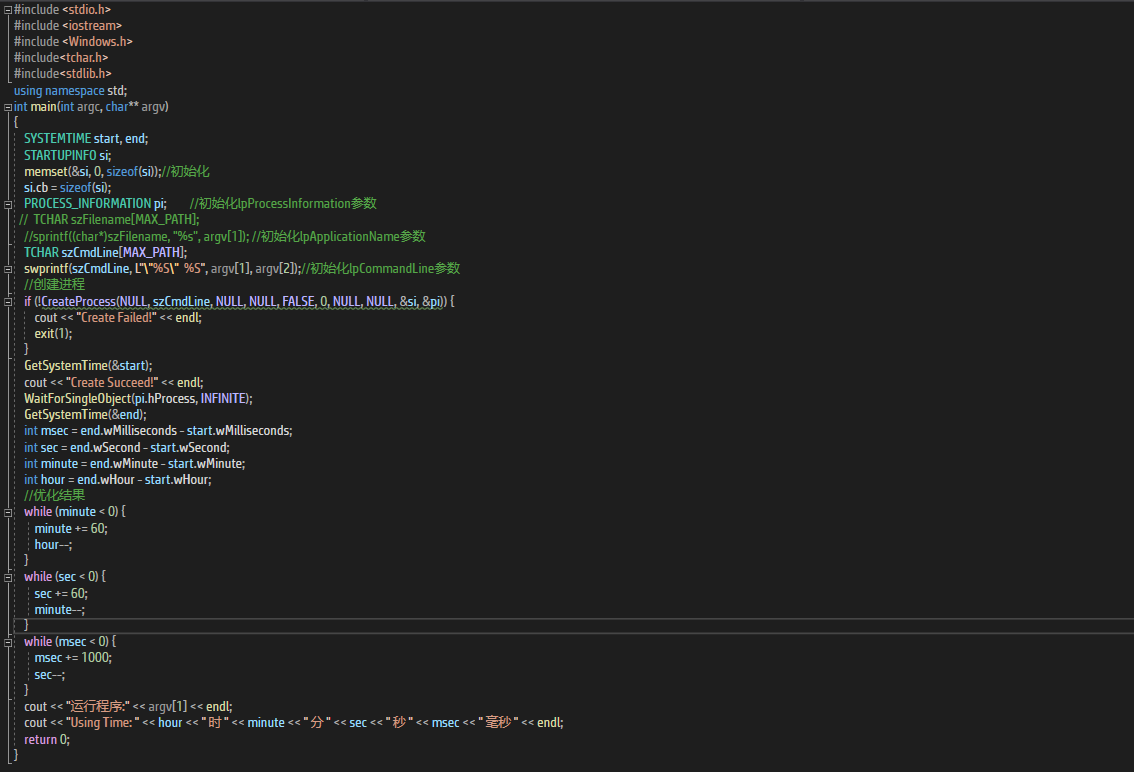
dwMilliseconds

超时间隔（以毫秒为单位）。如果指定了非零值，则函数将等待，直到发出信号通知对象或间隔过去为止。如果dwMilliseconds为零，则如果未用信号通知对象，则函数不会进入等待状态；否则，函数将进入等待状态。它总是立即返回。如果dwMilliseconds为INFINITE，则该函数仅在信号通知对象时返回。

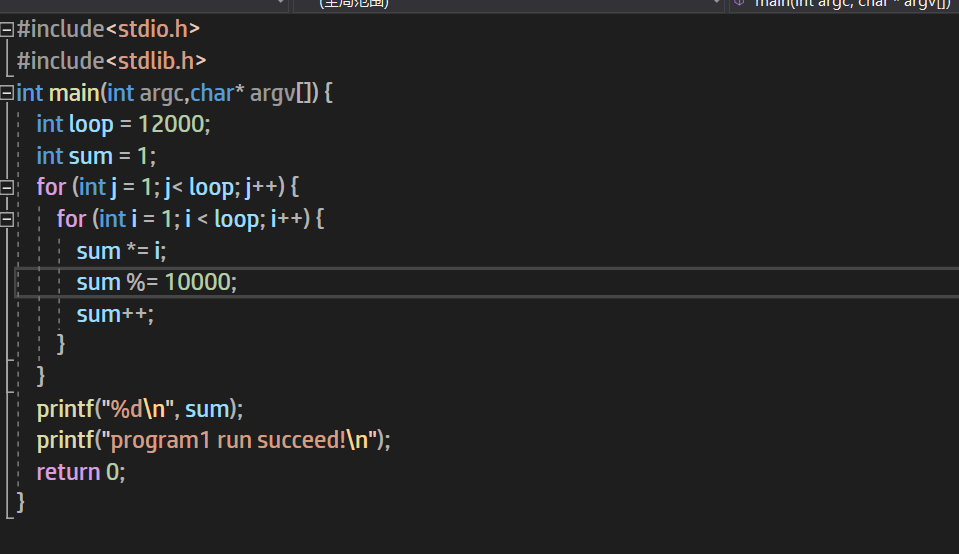
1. 使用GetSystemtime()进行计时，函数描述如下：

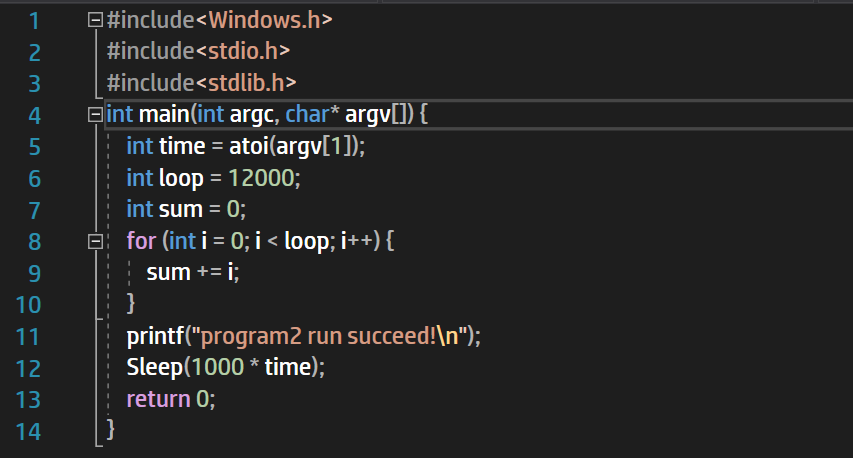
lpSystemTime

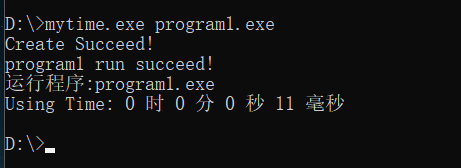
指向SYSTEMTIME结构的指针， 以接收当前系统日期和时间。该lpSystemTime参数不能为NULL。使用NULL将导致访问冲突

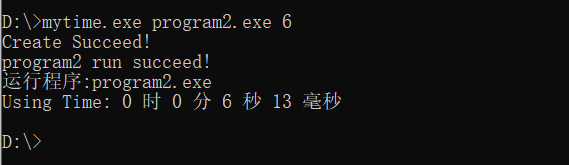
 Windows下mytime的源码：

program1的源码如下：



 program2的源码如下：

 运行program1成功的界面

 运行program2成功的界面

**在Linux下实现：**

1. 使用fork()创建进程运行程序

正确完成时，函数返回给父进程的是被创建子进程的标识，返回给子进程的为0；若创建失败，则返回父进程的为-1；

通过返回值，可以判断子进程是否创建成功，以及进程是子进程还是父进程（由于fork()函数调用后，子进程为父进程的一个拷贝）

1. 调用execv()函数为子进程指定运行程序

函数描述如下：

#include < unistd.h >

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

path为要指定的运行程序的源代码文件路径，argv[]为参数数组，返回值等于-1说明运行程序失败

1. 父进程使用wait()等待新创建的子进程结束，函数的具体说明如下

#include <wait.h>

int wait(int \*status)

父进程一旦调用了wait就立即阻塞自己，由wait自动分析是否当前进程的某个子进程已经退出，如果让它找到了这样一个已经变成僵尸的子进程，wait就会收集这个子进程的信息，并把它彻底销毁后返回；如果没有找到这样一个子进程，wait就会一直阻塞在这里，直到有一个出现为止

参数status是用来保存被收集进程退出时的一些状态，一般取NULL

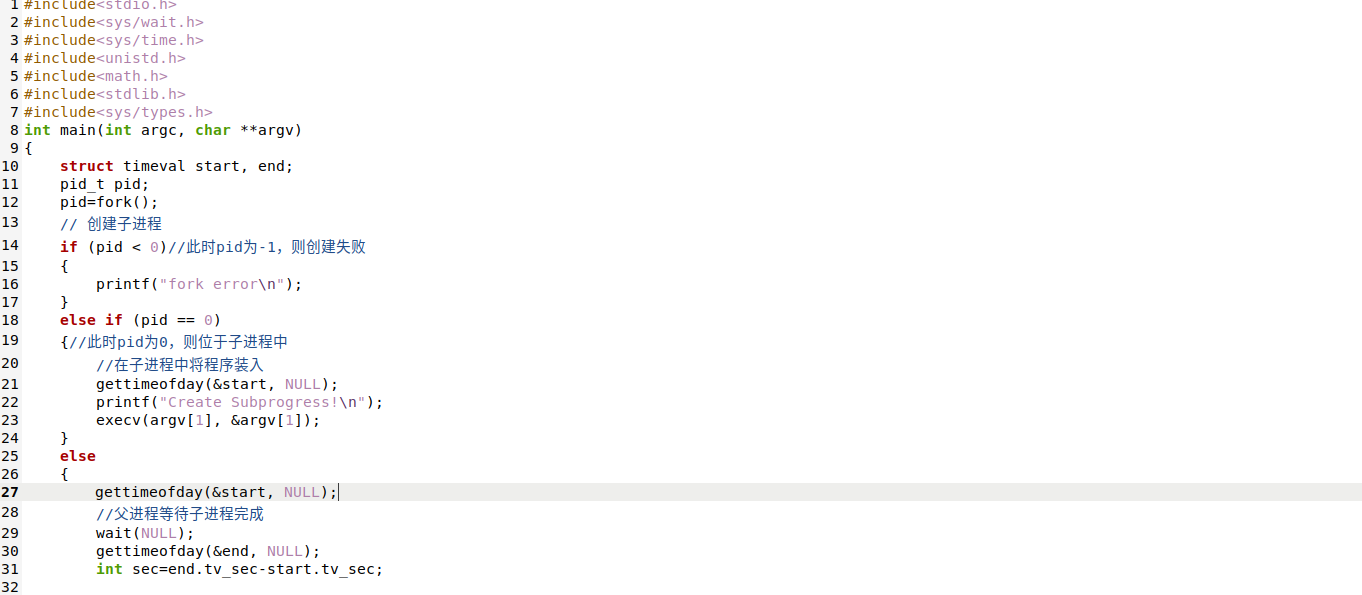
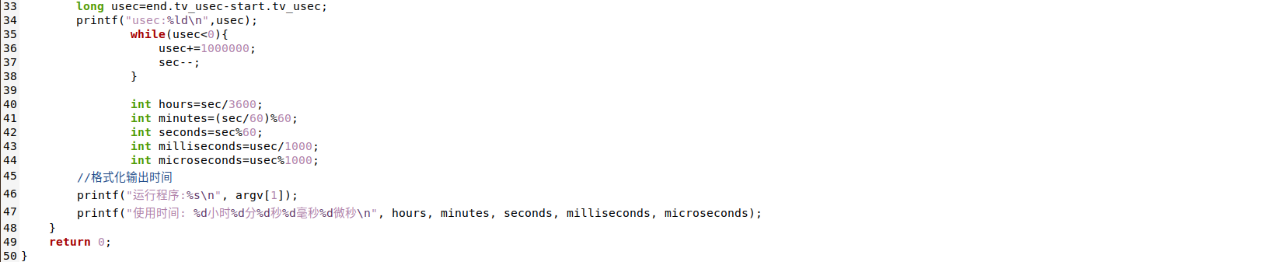
1. 调用gettimeofday()来获取子进程运行时间

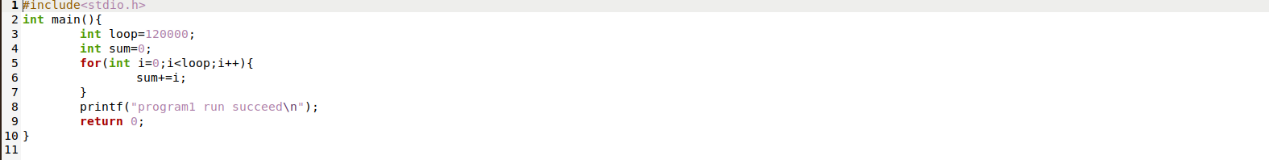
函数具体说明如下：

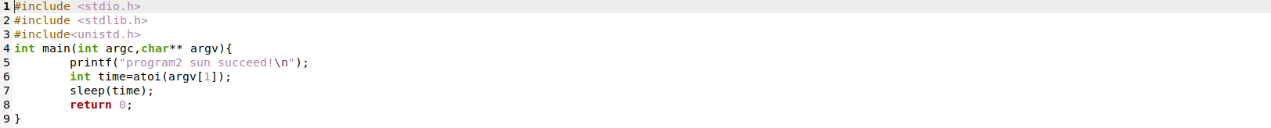
#include<sys/time.h>

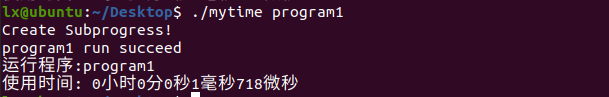
int gettimeofday(struct  timeval\*tv,struct  timezone \*tz )

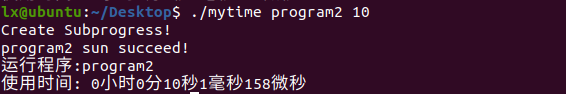
gettimeofday()会把目前的时间用tv 结构体返回，当地时区的信息则放到tz所指的结构中。在gettimeofday()函数中tv或者tz都可以为空。如果为空则就不返回其对应的结构体。函数成功执行后返回0，失败返回-1。

 Linux下mytime的源码如下：

 program1的源码如下：

 program2的源码如下：

 运行program1的结果：

 运行program2的结果：

1. **实验收获与体会**

学会了使用Linux和Windows系统下与进程控制相关的API，对比了两个操作系统的在进程控制上的不同。在linux中，在系统运行时，会产生一个进程号为0的进程，之后所有的子进程都是通过这个进程复制出来的，进程0一直处于空闲状态，并不占有cpu，所以进程之间是存在父子关系的。但是在Windows系统中，进程之间的关系比较松散，并不存在这样的父子进程。