

# 操作系统课程设计实验报告

实验名称： 进程控制

姓名/学号： 卜梦煜/1120192419

## 一、 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。

## 二、 实验内容

### 1. 在 Windows 下实现:

- 使用 `CreateProcess()`来创建进程。
- 使用 `WaitForSingleObject()`在“mytime”命令和新创建的进程之间同步。
- 调用 `GetSystemTime()`来获取时间。

### 2. 在 Linux 下实现:

- 使用 `fork()/vfork /exec()`来创建进程运行程序。
- 使用 `wait()`等待新创建的进程结束。
- 调用 `gettimeofday()`来获取时间。

### 3. 运行要求:

mytime 的用法: `$ mytime.exe program1`

要求输出程序 `program1` 运行的时间。`Program1` 可以为自己写的程序，也可以是系统里的应用程序。如 Linux: `ls vi top` 等命令 或者 Windows: `notepad`

`$ mytime.exe program2 t`

`t` 为时间参数，为 `program2` 的输入参数，控制 `program2` 的运行时间。最后输出 `program2` 的运行时间，应和 `t` 基本接近。

显示结果: `**小时**分**秒**毫秒**微秒`

## 三、 实验环境

Windows: Dev-C++ 5.7.1.0、命令行

Linux: Ubuntu 18.04.3、VSCode 1.62

## 四、 程序设计与实现

### 1. Windows 部分

#### (1) 程序设计

1) 创建变量，包括记录时间的变量、记录子进程启动信息的变量、记录子进程信息的变量。

2) 调用 `GetSystemTime()`记录此时系统时间，作为子进程启动时间。

3) 调用函数 `CreateProcess()`创建子进程。其中参数 `lpCommandLine` 为传入子进程的命令行内容，需根据参数个数直接传递或拼接后传递。

4) 调用等待函数，等待子进程结束。等待函数为 `WaitForSingleObject()`，其中等待时间设置为 `INFINITE`。

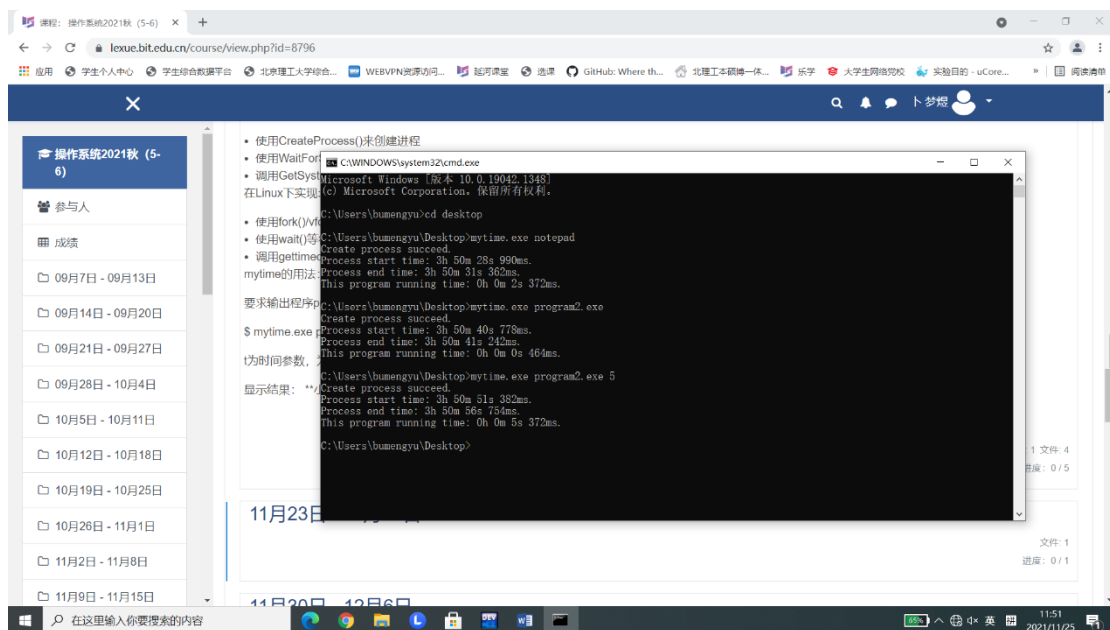
5) 子进程结束后，调用 `GetSystemTime()`记录子进程结束时间。

6) 第 5 步时间减去第 2 步时间即得到子进程运行时间。

#### (2) 运行结果

1) 进入.exe 文件所在路径。

2) 分别运行系统程序 `notepad`、自己编写的程序 `program2`。其中 `program2` 分别在指定时间参数和不指定时间参数的情况下运行。结果如下图。



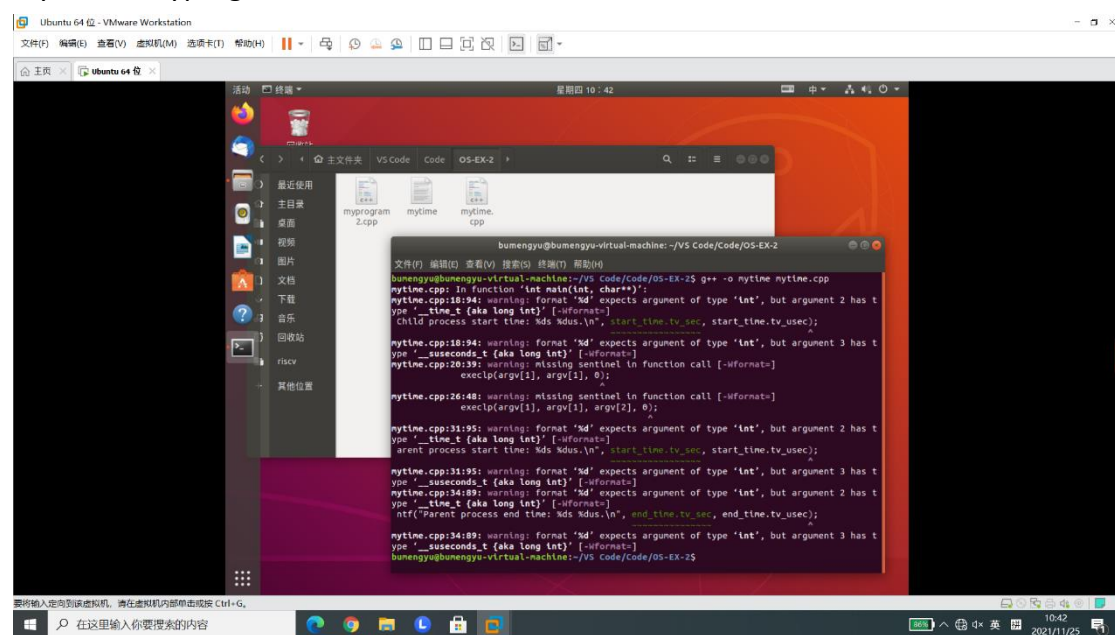
## 2. Linux 部分

### (1) 程序设计

- 1) 创建变量，包括记录时间的变量、记录子进程号的变量。
- 2) 调用 `gettimeofday()` 记录此时系统时间，作为子进程启动时间。
- 3) 调用函数 `fork()` 创建子进程，子进程调用 `execlp()` 函数切换至指定程序。
- 4) 调用等待函数 `wait()`，等待子进程结束。
- 5) 子进程结束后，调用 `gettimeofday()` 记录子进程结束时间。
- 6) 第 5 步时间减去第 2 步时间即得到子进程运行时间。

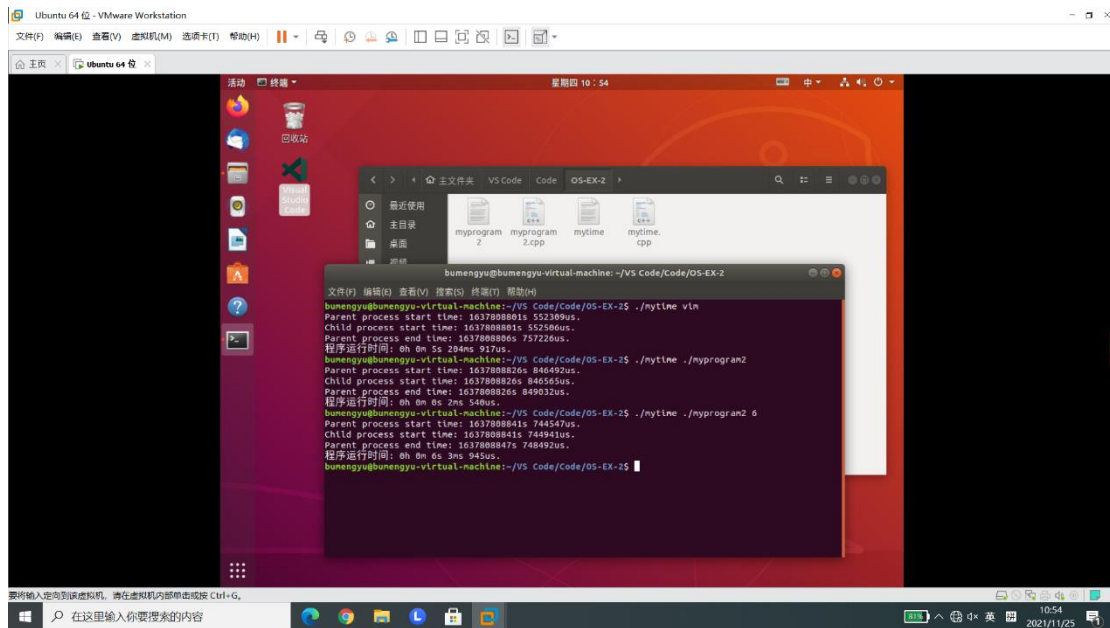
### (2) 运行结果

- 1) 调用命令 `g++ -o"将 mytime.cpp 和 myprogram2.cpp 编译成可执行文件 mytime、myprogram2。`



```
bumengyu@bumengyu-virtual-machine: ~/VS Code/Code/OS-EX-2
bumengyu@bumengyu-virtual-machine:~/VS Code/Code/OS-EX-2$ g++ -o mytime mytime.cpp
mytime.cpp: In function 'int main(int, char**)':
mytime.cpp:18:94: warning: format '%d' expects argument of type 'int', but argument 2 has type 'time_t (aka long int)' [-Wformat=]
Child process start time: %ds %dus.\n", start_time.tv_sec, start_time.tv_usec);
mytime.cpp:18:94: warning: format '%d' expects argument of type 'int', but argument 3 has type 'time_t (aka long int)' [-Wformat=]
mytime.cpp:20:39: warning: missing sentinel in function call [-Wformat=]
execlp(argv[1], argv[1], 0);
mytime.cpp:26:48: warning: missing sentinel in function call [-Wformat=]
execlp(argv[1], argv[1], argv[2], 0);
mytime.cpp:31:95: warning: format '%d' expects argument of type 'int', but argument 2 has type 'time_t (aka long int)' [-Wformat=]
Parent process start time: %ds %dus.\n", start_time.tv_sec, start_time.tv_usec);
mytime.cpp:31:95: warning: format '%d' expects argument of type 'int', but argument 3 has type 'time_t (aka long int)' [-Wformat=]
mytime.cpp:34:89: warning: format '%d' expects argument of type 'int', but argument 2 has type 'time_t (aka long int)' [-Wformat=]
ntf("Parent process end time: %ds %dus.\n", end_time.tv_sec, end_time.tv_usec);
mytime.cpp:34:89: warning: format '%d' expects argument of type 'int', but argument 3 has type 'time_t (aka long int)' [-Wformat=]
bumengyu@bumengyu-virtual-machine:~/VS Code/Code/OS-EX-2$
```

- 2) 分别运行系统程序 `vim`、自己编写的程序 `myprogram2`。其中 `myprogram2` 分别在指定时间参数和不指定时间参数的情况下运行。结果如下图。



## 五、 实验收获与体会

通过这次实验，我对 Windows、Linux 的一些 API 函数有了更深入的理解与使用。对函数的一些特性有了一些了解。

WaitForSingleObject()函数，第二个参数等待时间可指定为宏定义的 INFINITE，等待至子进程退出。

对 exec()函数族，选用 execlp()函数，第一个参数只需给出文件名，第二个参数为可变参数，根据命令行参数数量传递。终端调用自己写的程序时，命令格式为“./mytime ./myprogram2 5”。