**操 作 系 统**

**实 验 报 告**

**实验名称：实验二 进程间通信和命令解释器**

**姓名： 陈衍斌**

**学号： 16340042**

实验名称：进程间通信和命令解释器

一、实验目的：

1. 进程间共享内存实验，初步了解这种进程间通讯
2. 实现简单的shell命令解释器：了解程序运行。

二、实验要求：

1. 进程间共享内存实验：完成课本第三章的练习3.10的程序。

2. 完成课本上第三章的项目：实现shell。除此之外满足下面要求：

* + 实现程序的后台运行

三、实验过程：

1. 课本3.10要求实现在子程序输出Fibonacci序列。

课本上已经给了共享内存段的数据结构的定义。下面先介绍一些需要用到的函数：

**shmget**：创建或打开共享存储区。依据用户给出的整数值key，创建新区或打开现有区，返回一个共享存储区ID。

调入shmget函数时，传入的key不同，就能产生不同的共享内存。

**shmat：**连接共享存储区

**shmdt**：拆除共享存储区连接

**shmctl**：共享存储区控制

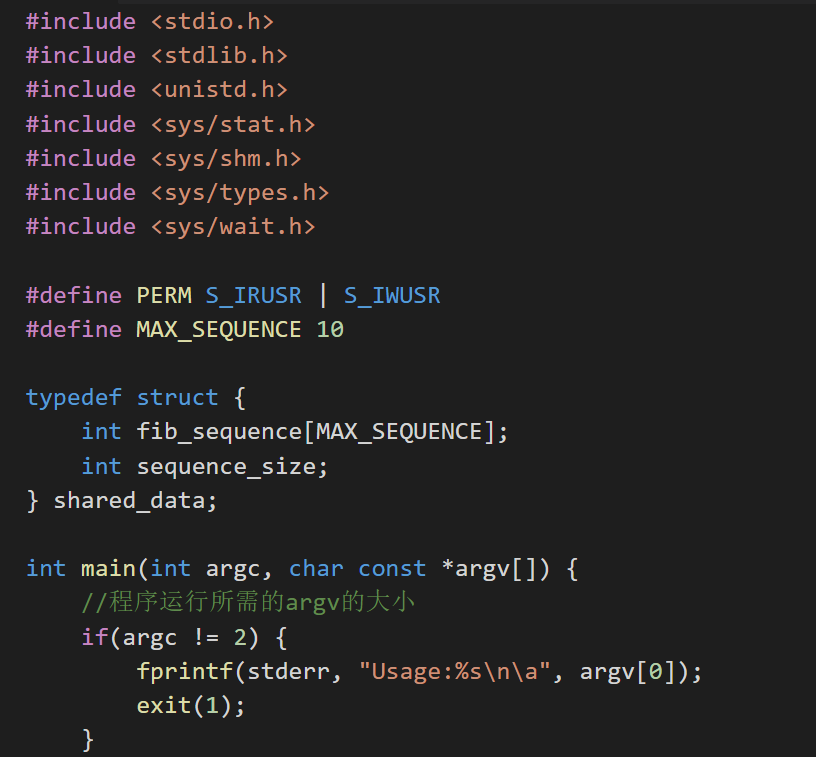
**IPC\_PRIVATE：**保证使用唯一ID(或key)

**S\_IRUSR | S\_IWUSR：**使当前用户可以读写这个区域

接下来简要分析一下实现的过程：

* 首先，共享空间的结构，课本上已经定义了，包含一个存储斐波那契数列的数组和一个保存长度的变量。
* 判断程序中输入的合法性。这需要判断输入的参数个数以及参数大小范围是否合法。
* 分配共享空间，获取共享内存的地址。
* 创建子进程，子进程输出的数列存储在共享内存中。
* 在父进程中将共享内存中的斐波那契数列输出，输出结束后断开连接，然后删除共享内存

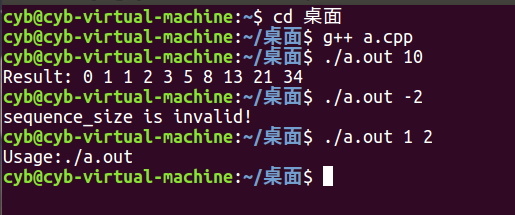
其他没什么好说的，下面贴上代码（一些代码的解释见注释）：







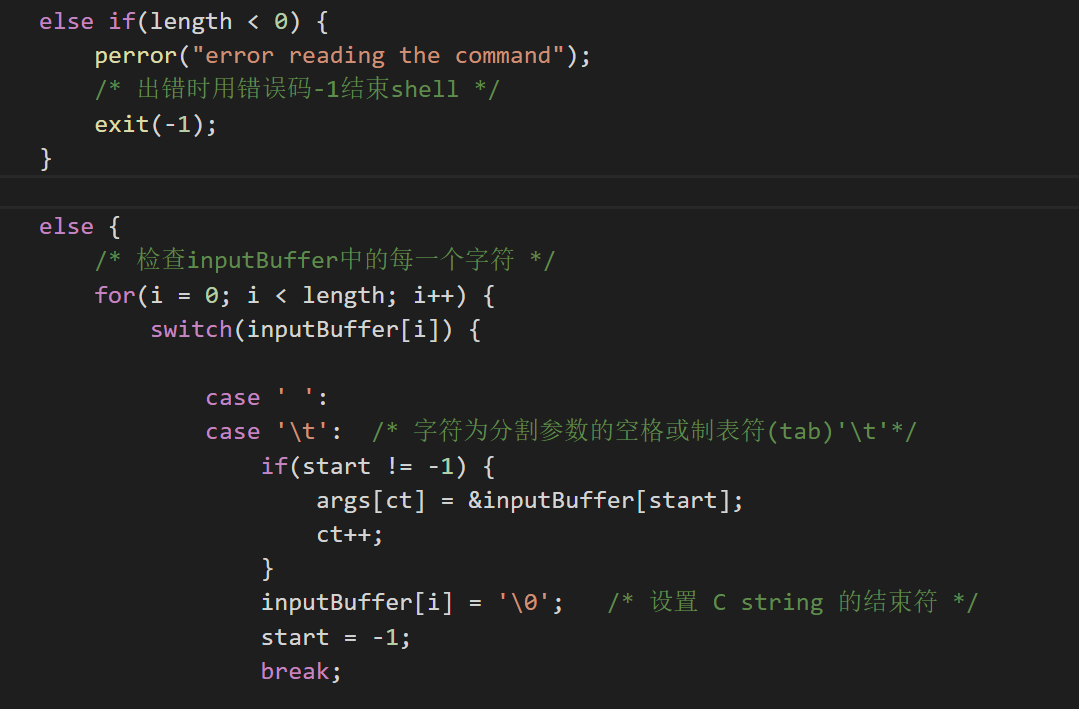
输出如下：



2. 课本中给的Shell程序的基本框架包含两个函数：main()和setup()。setup()函数用于读取用户的下一条命令（最多80个字符），然后将之分析为独立的标记，这些标记被用来听啊重命令的参数向量（如果将要在后台运行命令，它将以“&”结尾，setup()将会更新参数background，以使main()函数相应地执行）。当用户按快捷键Ctrl+D后，setup()调用exit()，此程序被终止。

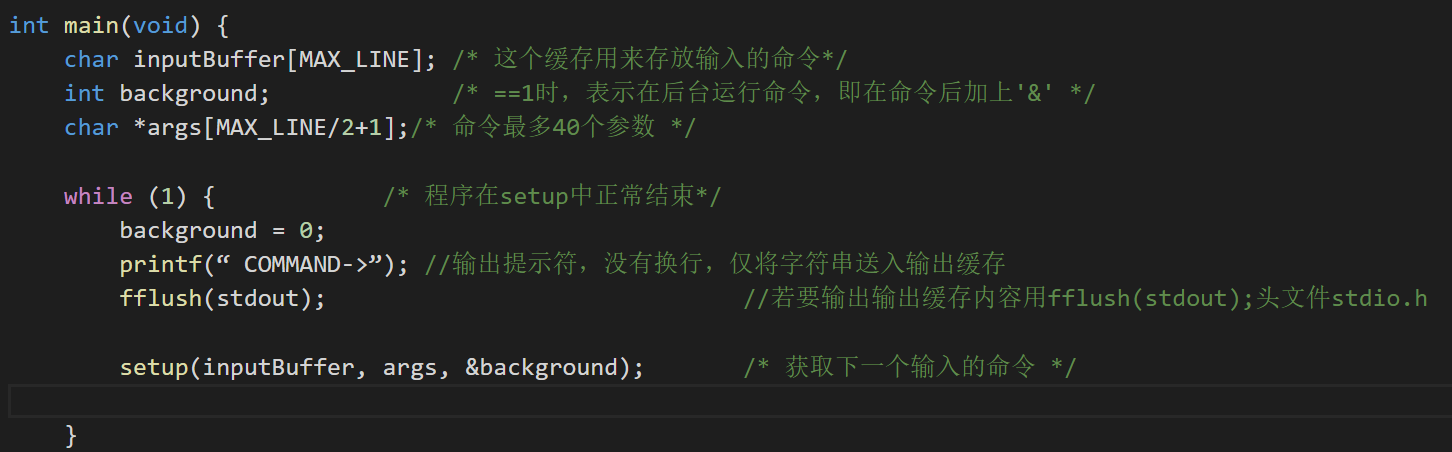
setup()函数地代码如下（ppt上面其实已经提供了）：







接下来时main()函数，main函数实现的基本框架ppt和课本都已经给了出来：

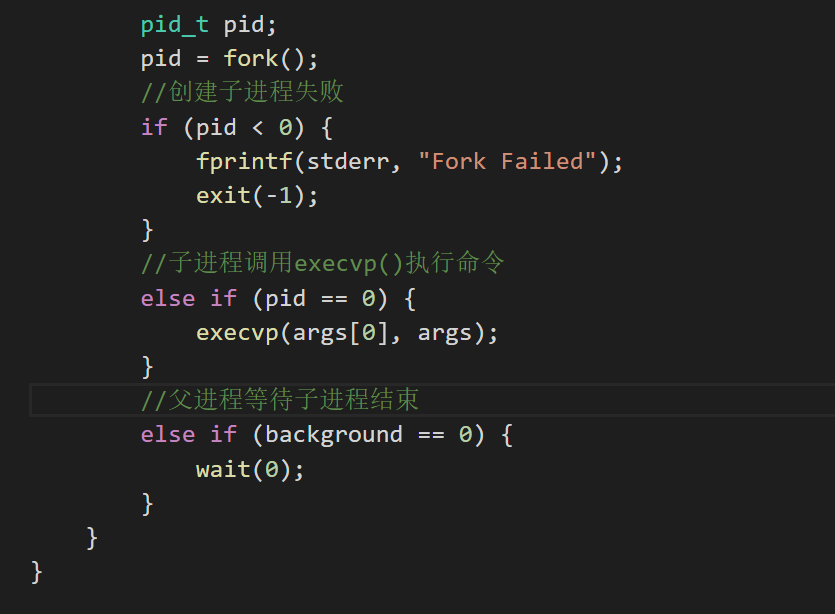


我们所需要补充的是while(1)循环中的内容。首先我们通过setup函数读取了用户输入的指令并将其切割存放在到了数组中。接下来根据ppt，我们需要完成的任务是：

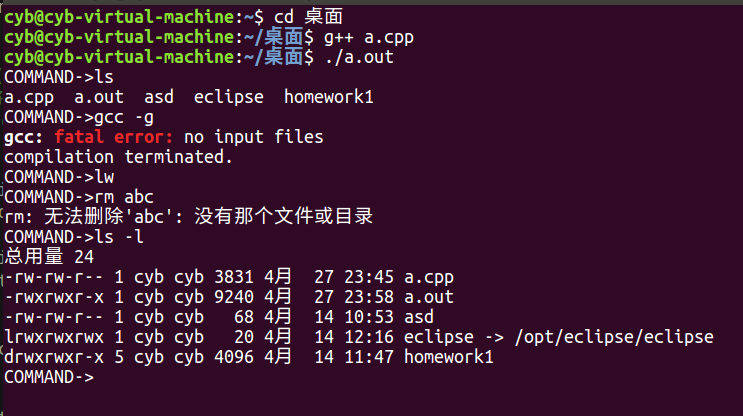
* 用fork()产生一个子进程；
* 子进程将调用execvp()执行命令,即execvp(args[0],args)；
* 如果background == 0, 父进程将等待子进程结束, 即if(background==0) wait(0)；否则，将回到函数setup()中等待新命令输入。

fork()函数和子进程与父进程的关系在上一个实验其实已经基本明确了，这次新接触的是execvp()函数。函数原型为execvp(char cmd[], char \* args[])；它的作用是从PATH 环境变量所指的目录中查找符合参数file 的文件名, 找到后便执行该文件, 然后将第二个参数argv传给该欲执行的文件。如果执行成功则函数不会返回, 执行失败则直接返回-1, 失败原因存于errno 中。

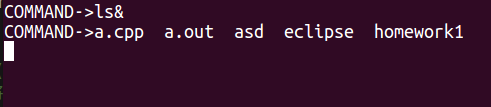
其他的实在是没什么好说的（毕竟这次的要求只需要实现程序的后台运行，像课本上实现Ctrl+c，历史记录什么的都不需要），所以就直接贴代码吧：



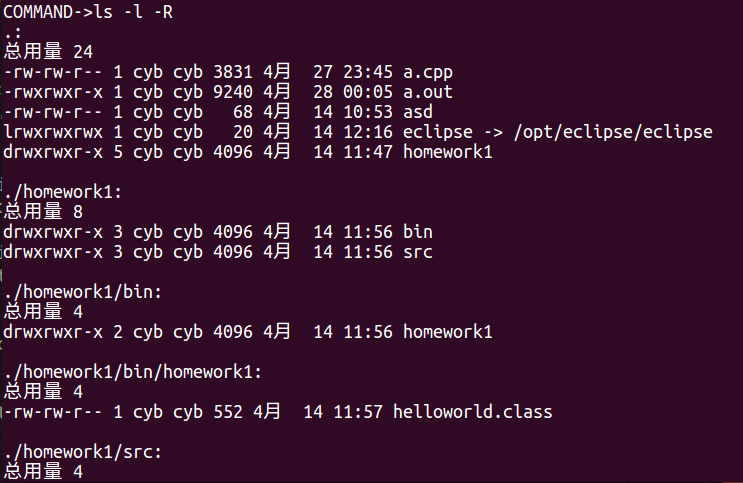
shell的具体实验结果如下：

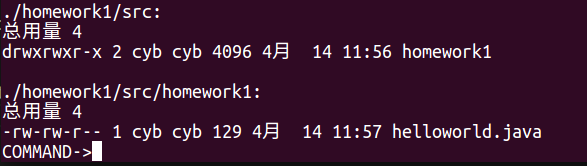


在输入的命令后面加上&，可以使命令在后台运行(光标停止在下一行，不会出现COMMAND->)：

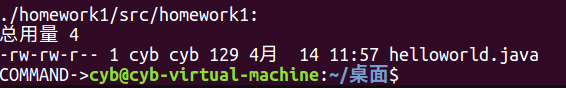


若键盘输入的命令为：ls -l –R，输出如下：





输入Ctrl+D，可以退出shell：



总结：这次实验总体的难度并不大，只要认真看课本和ppt都是比较简单的。第一个实验只要思路清晰就容易实现，第二个实验setup()函数的关键代码ppt里面都已经给出来了，这就降低了这次实验的难度。总体而言，通过这次实验，我对共享内存和shell的实现有了初步的了解。与此同时，对进程生成的过程和工作原理有了进一步的巩固，收益匪浅。