操作系统实验报告

实验名称	实验 2：生产者消费者问题		
学号	1120220576	姓名	宋宇翔

一、实验目的

1. 生产者与消费者

生产者与消费者，是进程管理中的一种抽象概念，分别通过产生数据以及接受数据来实现进程之间的通讯，从而有效地协调不同进程之间的分工与合作。

2. 进程的创建与 PV 操作的实现

在实际的操作系统如 Linux 中，系统的进程创建、通信、PV 操作都需要特定的系统库函数，如 fork()等，因此该实验也意在学习在不同系统下的并行编程。

二、实验内容

分别在 Linux 系统、Windows 系统中创建 3 个生产者、4 个消费者，并不断向长度为 4 的缓冲区中放入名字首字母（本实验中从“SYX”中随机选择一个字母），每次操作之间随即等待 3s 以内的时间，分别重复 4 次、3 次。每次操作输出插入或读取的信息以及缓冲区的状态。

三、实验环境及配置方法

Linux 系统，Windows 系统。

四、实验方法和实验步骤（程序设计与实现）

首先，生产者与消费者之间的 PV 操作，可以参考生产者和消费者问题中的互斥逻辑，即设立三个互斥信号量：empty、full 和 mutex。其中 empty 表示缓冲区的空缓冲数量，用于限制生产者向缓冲区放入数据的数量，在这里初值为 4；full 与之相反，表示缓冲区中已使用的缓冲个数，用于防止消费者读取空缓冲；mutex 为访问临界区的互斥信号量，保证任意时刻只能有不超过一个的进程在访问缓冲区。

在 Linux 系统中创建进程需要进行以下步骤：

```
1. pid_t pid=fork();
```

而在 Windows 中需要以下代码：

```
1. TCHAR filename[MAX_PATH];
2. GetModuleFileName(nullptr, filename, MAX_PATH);
3.
4. TCHAR cmdLine[MAX_PATH];
```

```
sprintf(cmdLine, "\"%s\" %d", filename, type);

STARTUPINFO si = {sizeof(STARTUPINFO)};
PROCESS_INFORMATION pi;

//新建子进程
BOOL bCreateOK = CreateProcess(
            filename,
            cmdLine,
            nullptr,
            nullptr,
            FALSE,
            CREATE_DEFAULT_ERROR_MODE,
            nullptr,
            nullptr,
            &si,
            &pi);

```

由于两系统中的操作逻辑一致，因此以下主要展示生产者消费者中的核心代码。

生产者部分核心代码：

```
P(semid,EMPTY);//P
P(semid,MUTEX);
usleep(randMod(3e6));//随机等待

strncpy(shmptr->str[shmptr->tail],randString(),BUF_LEN);//写入
printf("[pid %d] push %-s ",getpid(),shmptr->str[shmptr->tail]);

shmptr->tail=(shmptr->tail+1)%BUF_CNT;
for(int j=0;j<BUF_CNT;j++)//输出当前缓冲区状态
{
    printf("|%c",shmptr->str[j][0]);
}
printf("|\n");
//fflush(stdout);//清空输出缓冲
V(semid,FULL);//V
V(semid,MUTEX);
```

可见在此程序中，仿照生产者与消费者问题的执行逻辑（如图所示）

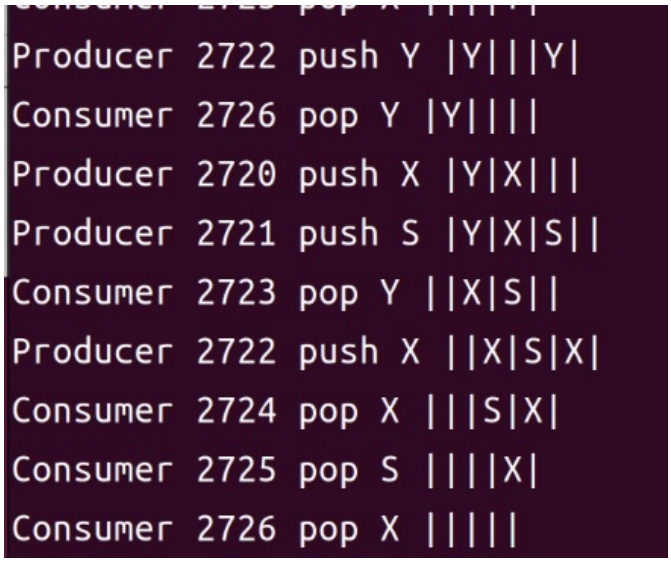
```
生产者进程（Producer）：
  ...
  produce a product x;
  P(empty);                 //申请一个空缓冲
  P(mutex);                 //申请进入缓冲区
  array[i] = x;             //放产品
  i = (i+1)mod k;           //修改写指针
  V(full);                  //有数据的缓冲区个数加1
  V(mutex);                 //退出缓冲区
```

消费者部分的代码在此省略。

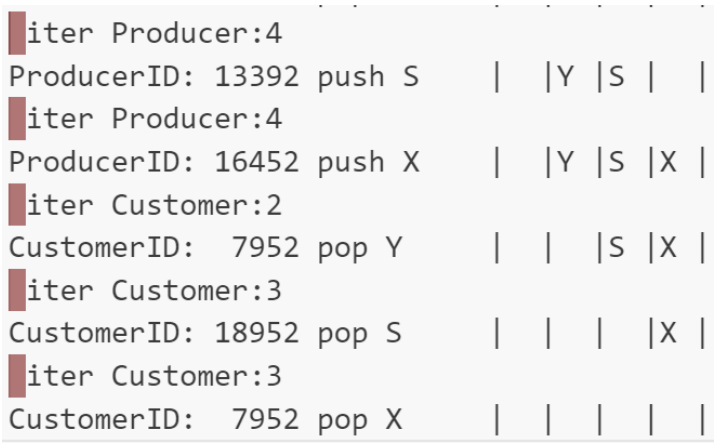
需要注意的是，在 Windows 操作系统中，需要调用 windows.h 库，同时通过 CreateProcess()函数进行进程创建，而非 Linux 系统中的 fork()。

五、实验结果和分析

在 Linux 系统终端用 gcc 编译器进行编译后运行，得到如下输出：



Windows 运行结果如下：



可见在以上的两个运行结果中，虽然生产者与消费者的运行顺序不尽相同，但都保证了任意时刻只有一个进程在访问缓冲区，每一进程都按照自己的访问顺序访问缓冲区，同时在程序运行终止时，缓冲区为空（由于生产者与消费者各填入和删除 12 次数据）。

六、讨论、心得

对于 Linux 系统，线程编程相对较简单，使用 fork()申请进程后只需记录进程编号，需要注意的是主程序中进程创建和执行的顺序和逻辑。

而在 Windows 系统中创建进程，需要声明一系列变量，如安全属性，命令行字符串等，在编写创建进程代码时需要注意。

七、注意事项：

17. 报告命名：

实验次数+学号+姓名，如 1-201107302-某某某.doc，（用半角字符）。

18. 附实验要求（每个实验共 15 分）

1. 实验 1
   19. 报告内容完整、步骤结果详实、条理清楚
   20. 报告整洁（排版，错别字等）
   21. 迟交扣 2 分
2. 实验 2，3，4
   22. 代码能正确运行、功能完整
   23. 代码整洁（注释，缩进等规范）
   24. 报告内容完整、步骤结果详实、条理清楚
   25. 报告整洁（排版，错别字等）
   26. 迟交扣 2 分
3. 全部实验
   对抄袭视程度扣分 3-5 分，不能判定抄袭和被抄袭者，两者全扣