### 实验2 生产者消费者问题

班级： 07152001 学号： 1120201198 姓名： 史桠彬

一、实验目的

通过本实验，加深对进程概念的理解，明确进程与程序的区别；认识并发执行的本质，理解和掌握Linux和Windows进程通信系统调用的功能，通过实验和学习，提高对进程痛惜系统调用的编程能力；掌握使用信号量机制完成进程间同步和互斥的方法。

二、实验内容

在Windows操作系统下创建一个有6个缓冲区的缓冲池，初始为空，每个缓冲区能存放一个长度为10个字符的字符串，同时创建2个生产者和3个消费者，用进程模拟生产者或消费者。

对于生产者而言，随机等待一段时间后，往缓冲区中存数据，若缓冲区满，则等待消费者取走数据后再存放，每个生产者存12次数据。

对于消费者而言，随机等待一段时间后，从缓冲区中取数据，若缓冲区空，则等待生产者存放数据后再取走，每个消费者取8次数据。

显示每次向缓冲区添加或取走的数据以及时间，每次操作后都要显示缓冲区中的全部数据，生产者、消费者进程号。

三、程序设计与实现

**1. 实验环境**

虚拟机： CentOS 7  
操作系统：Windows10  
开发环境：RedPanda-Cpp.0.14.2.64位.10.3

1. **设计思路**

① 定义共享内存区

struct BUF {

    char **array**[BufAmount][BufLen];

    int **head**;

    int **tail**;

    int **IsEmpty**;

};

共享内存区中的array表示生产者和消费者存取数据的环形缓冲区，设置其大小BufAmount为6，其单条数据长度BufLen为10。指针head用于指向消费者下一次待取的数据，指针tail指向生产者下一次需要存放数据的缓冲，IsEmpty用来表示环形缓冲区是否为空，取0为不空，取1为空。

② 设置信号量

在实验中，设置三个信号量如下：

互斥信号量**MUTEX**：用于生产者进程与生产者进程、消费者进程与生产者进程、消费者进程与消费者进程间互斥使用缓冲区，初始值为1。

同步信号量**EMPTY**：用于指示当前空缓冲区的可用数量，用于制约生产者进程向缓冲区存数据，初始值为6。

同步信号量**FULL**：用于指示当前有数据的缓冲区的数量，用于制约消费者进程取数据，初始值为0。

③ 生产者存数据

(1) GetRandomChar()和GetRandomNum()：随机产生不定长字符串作为数据。

(2) Sleep(GetRandomSleep())：随机进行等待。

(2) P(EMPTY)：申请一个空缓冲单元，EMPTY-1。

(4) P(MUTEX)：申请环形缓冲区的唯一使用权，MUTEX-1。

(5) strcpy(pbuf->array[pbuf->tail],charray)：生产者进程向缓冲区中存放数据。

(6) pbuf->tail=(pbuf->tail+1)%BufAmount：修改尾指针，指向下一个缓冲单元。

(7) pbuf->IsEmpty=1：修改缓冲区状态IsEmpty。

(8) V(MUTEX)：生产者进程释放对环形缓冲区的使用权，MUTEX+1

(9) V(FULL)：FULL+1，表示环形缓冲区中可取数据的总量，唤醒消费者进程

④ 消费者取数据

(1) Sleep(GetRandomSleep())：随机进行等待。

(2) P(FULL)：申请一个放有数据的缓冲单元，FULL-1。

(3) P(MUTEX)：申请环形缓冲区的唯一使用权，MUTEX-1。

(4) strcpy(charray,pbuf->array[pbuf->head])：取得当前头指针指向的数据，(5) memset(pbuf->array[pbuf->head,‘\0’, sizeof(pbuf->array[pbuf-> head])) ：清除存储这一数据的缓冲单元的内容。

(6) pbuf->IsEmpty=(pbuf->head==pbuf->tail)：修改缓冲区状态.

(7) pbuf->head=(pbuf->head+1)%BufAmount：修改指针指向下一缓冲单元。

(8) V(MUTEX)：消费者进程释放对环形缓冲区的使用权.

(9) V(EMPTY)：Empty+1，表示环形缓冲区中空缓冲单元的数量，用于唤醒生

产者进程向缓冲区中存数据。

在生产与消费的过程中，有同步信号量FULL和EMPTY控制消费者或生产者是否能够申请缓冲单元，避免出现缓冲区满而生产者仍然往里放数据，缓冲区空而消费者仍然在取数据的情况。

此外，由于MUTEX的存在，保证各进程能够互斥使用环形缓冲区，MUTEX、FULL与EMPTY的合理搭配保证不会有死锁的产生。

1. **编写、编译代码**

在开发环境中编写C++代码，调用相关接口实现相应功能。

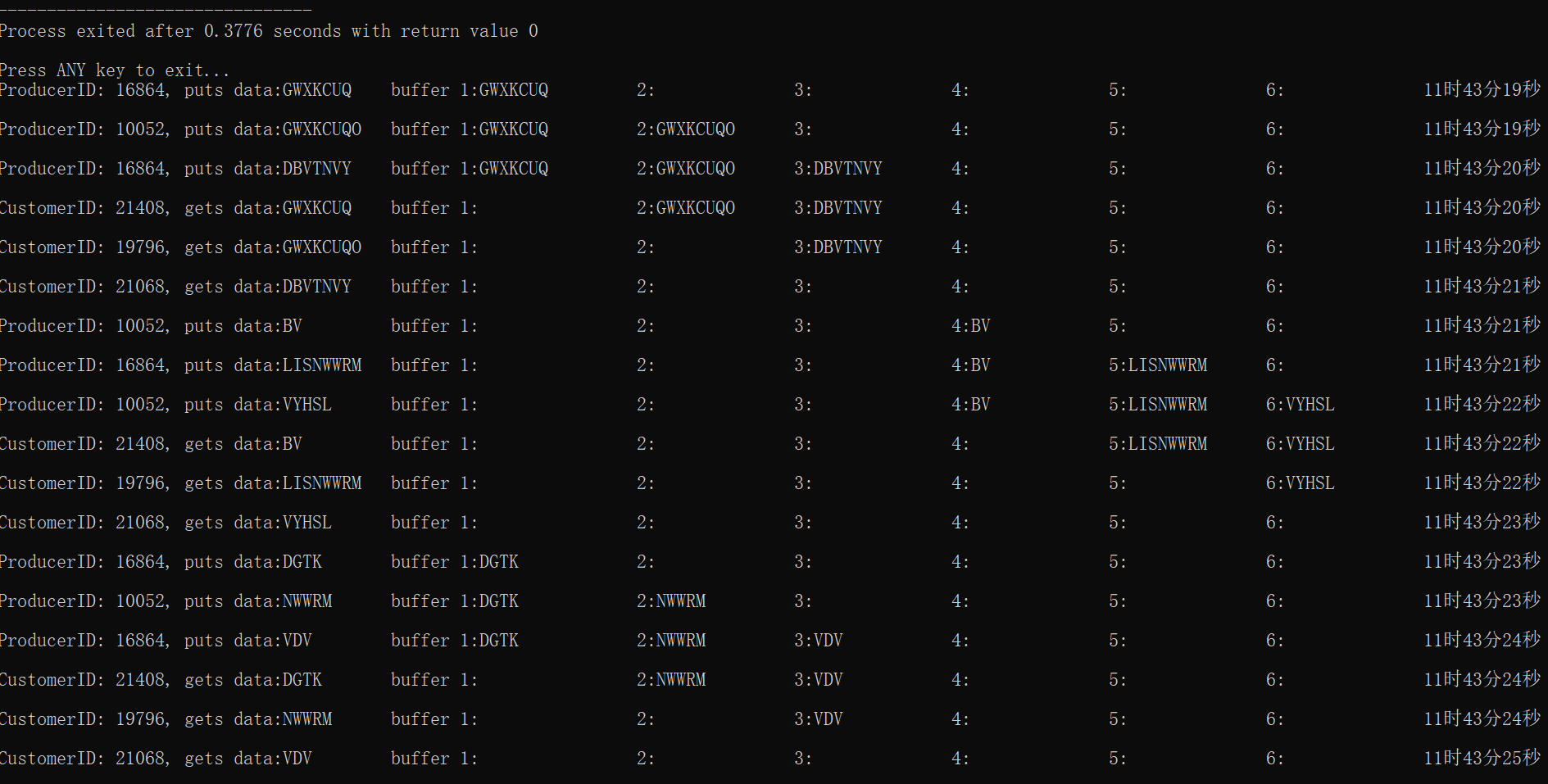
1. **运行得到结果**

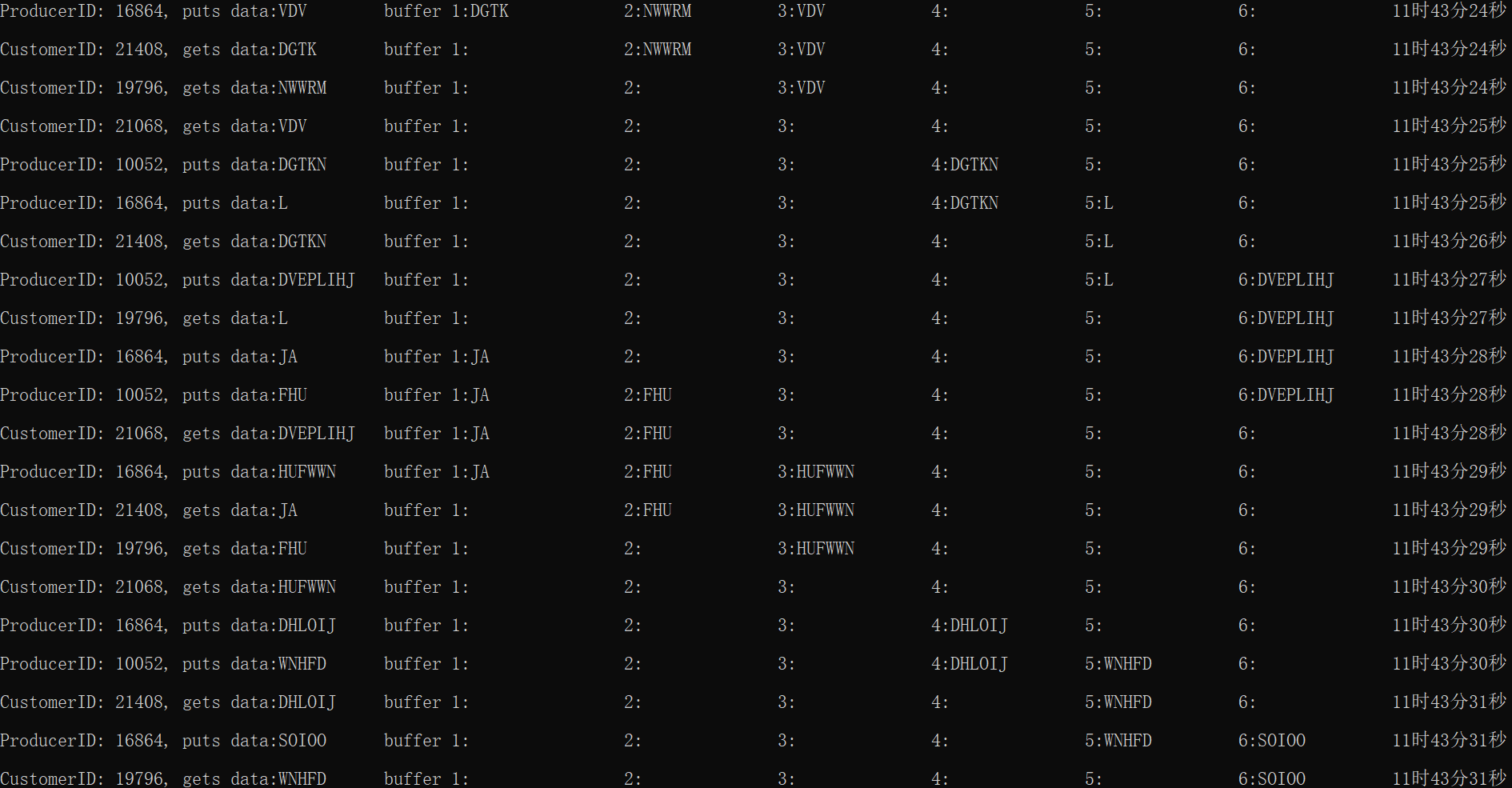
四、实验结果及分析

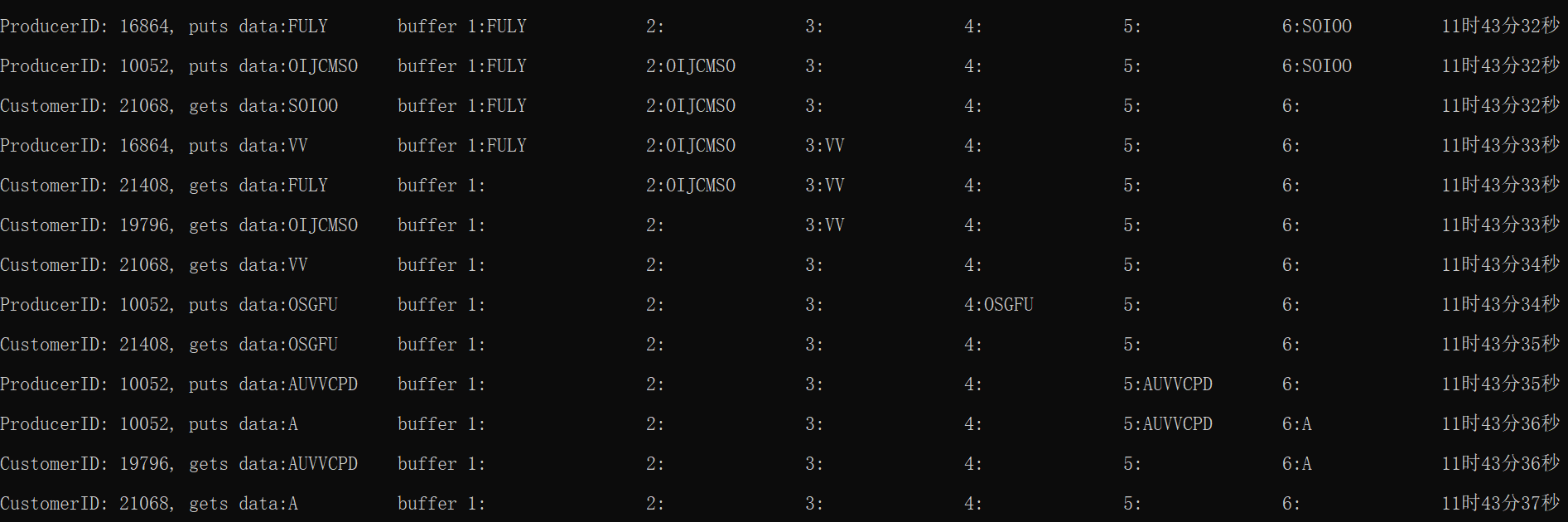
操作过程中，通过进程模拟2个生产者，编号分别为16864、10052，3个消费者，编号为21408、19796、21068。

从2022年10月5日11时43分19秒到11时43分37秒，父进程共完成24次存数据和24次取数据过程，环形缓冲区由空到满再次到空，实现字符串数据的存取，“生产者消费者问题”得到解决。

实验结果如下图。







五、实验收获与体会

“生产者消费者问题”是信号量知识里非常经典的案例，通过本次实验，我进一步加深了对进程概念的理解，深刻认识到并发执行的本质。

在Windows操作系统里，我利用信号量机制用进程模拟生产者和消费者，完成了进程间的同步和互斥。在看到正确运行结果的那一刻，我觉得编写与修改代码所付出的一切辛苦都是值得的。

在亲自动手实验的过程中，我对信号量知识点有了更深刻的理解，同时也复习了程序设计方面的知识，真可谓“一举两得”，这次实验让我收获颇多！

附录：程序清单及说明

仅有源代码Codes.cpp。