OS 第6次作业PV操作题

2021.5.28修改

根据OS第6次作业中的信号量及PV操作题原答案的错误，以及同学们答题的情况，整理了本文档。

文档中除了原作业题的几种新答案之外，还引申出了3个形式上类似，但难度逐步增加的题目，并在最后摘录了其他试卷上的3个同类题目及附属答案，共7个题目。

撰写整理本文档的目的，除了更正原答案的错误外，更是希望同学们通过阅读学习本文档，进一步掌握PV操作中解决问题的办法。学习如何利用问题的等效性，套用已掌握的代码。以及在不改变原问题的前提下，把PV操作的过程划分为容易解决的子过程，逐个击破。

所有答案仅供参考，不敢说一定正确，当然更不能说是唯一答案。请同学们带着批评的眼光阅读研究，并欢迎指正。

**题目A：**

某客运轮渡码头对乘客进行新冠检查。有一80人的休息室供乘客等候检查，开始的时候休息室为空。乘客在休息室有空位时进入休息室等候检查，否则在休息室外等待。仅有一名医生，每次呼叫一个在休息室中等待的乘客进行检查，无乘客时医生休息。试用信号量及PV操作描述乘客及医生的活动。

**答案A0：**(乘客进程有错误)

int CHAIRS = 80; // 为乘客准备的椅子数

int waiting = 0; // 休息室等候乘客数

semaphore passengers = 0, doctor = 0, mutex = 1, ext\_wait = 0;

doctor() {

while(true) {

P(passengers); // 检查完一人,还有乘客吗？若无乘客,医生休息

P(mutex); // 进程互斥

waiting--; // 休息室等候人数减1

V(doctor); // 医生叫一个乘客检查

V(mutex); // 开放临界区

V(ext\_wait); // 唤醒门外乘客,若有

Check(); // 检查中

}

}

passenger() {

P(mutex); // 进程互斥

if(waiting < CHAIRS) { // 有无空椅子

waiting++; // 等候乘客数加1

V(passengers); // 有可能唤醒医生

V(mutex); // 开放临界区

P(doctor); // 医生忙,乘客坐下等待

Being\_checked(); // 一个乘客被检查

}

else {

V(mutex)

P(ext\_wait); // 人满了，在门外等待

}

}

在答案A0的乘客进程中，若乘客到达休息室时没有空椅子，则进入else语句块中，释放互斥锁后，在门外等待。当该乘客从门外等待状态被唤醒后，**不屑一顾地径直离去了！**根本没有进入休息室！

另外doctor已用作过程名，又用作信号量名，不严谨。

把常数CHAIRS定义为int型，而又未明确注释说明是常量，容易让他人误以为是想定义共享变量。

该题是理发店问题的轻微变形，OSC7书中习题6.11，答案在我们的ppt中有。与理发店不同的是，题目A中的乘客休息室满了，乘客会在室外等待，而不是理发店问题中的离去。

**答案A1：**

#define CHAIRS 80 // 为乘客准备的椅子数

int waiting = 0; // 休息室等候乘客数

semaphore passenger = 0, sdoctor = 0, mutex = 1, ext\_wait = 0;

doctor()

{

while(true) {

P(passenger); // 检查完一人,还有乘客吗？若无乘客,医生休息

P(mutex); // 进程互斥

waiting--; // 休息室等候人数减1

V(sdoctor); // 医生叫一个乘客检查

V(mutex); // 开放临界区

V(ext\_wait); // 唤醒门外乘客,若有

Check(); // 检查中

}

}

passenger() // 为每一位乘客启动一个乘客进程

{

while(true) {

P(mutex); // 进程互斥

if(waiting < CHAIRS) { // 有无空椅子？

waiting++; // 等候乘客数加1

V(passenger); // 有可能唤醒医生

V(mutex); // 释放临界区

P(sdoctor); // 医生忙,乘客坐下等待

Being\_checked(); // 一个乘客被检查

break; // 检查完，离开

}

else {

V(mutex) // 释放临界区

P(ext\_wait); // 人满了，在门外等待；被唤醒时重新回到进程开始处

}

} // while(true)

}

注：“Check()”与“Being\_checked()”，是双方一起参与的同一个过程段，只是两种描述的视角不同，因此一定同时发生，同时结束。

**答案A2：**

semaphore nChair = 80; // 空椅子数

semaphore nWait = 0; // 休息室中等待乘客数

semaphore sDoctor = 0; // 乘客与医生同步的信号量

doctor()

{

while(true) {

P(nWait); // 检查休息室中是否有乘客，无则医生休息

V(sDoctor); // 唤醒一名休息室中等待医生的乘客，叫入检查室检查 **S1**

V(nChair); // 空椅子数+1，有可能唤醒室外等待的乘客 **S2**

Check(); // 医生检查中

}

}

passenger() // 为每一位乘客启动一个乘客进程

{

P(nChair); // 休息室有空椅子则进入，否则在室外等待

V(nWait); // 在休息室等待医生的乘客数+1，有可能唤醒医生

P(sDoctor); // 等待医生

Being\_Checked(); // 乘客被检查中

// 检查完，离开

}

注意医生进程中语句**S1**与**S2**的顺序。顺序颠倒的话，会导致在尚未唤醒一名休息室中的乘客离开座椅之前，另一名乘客就可能从室外进入休息室，那样休息室中就超过80个人了。

**答案A3：**

semaphore nChair = 80; // 空椅子数

semaphore nWait = 0; // 休息室中等待乘客数

semaphore mutex = 1; // 乘客(们)占用医生的互斥信号量

doctor()

{

while(true) {

P(nWait); // 检查休息室中是否有乘客，无则医生休息

V(nChair); // 空椅子数+1，有可能唤醒室外等待的乘客

Check(); // 医生检查中

//V(mutex); // 医生检查完一名乘客，有可能唤醒休息室等待的乘客 **S3**

}

}

passenger() // 为每一位乘客启动一个乘客进程

{

P(nChair); // 休息室有空椅子则进入，否则在室外等待

V(nWait); // 在休息室等待医生的乘客数+1，有可能唤醒医生

P(mutex); // 等待医生

Being\_Checked(); // 乘客被检查中

V(mutex); // 释放医生互斥锁，有可能唤醒休息室等待的乘客 **S4**

// 检查完，离开

}

注：语句**S3**与**S4**中仅存在一条，两种实现等效。

**答案A4：**（由张凌霄同学提供）

semaphore room = 80; // 代表休息室的剩余位置

semaphore doc = 1; // 代表剩余可行动的医生

semaphore nwait = 0; // 代表现有的等待的乘客数目

每一个乘客()

{

wait(room); // 进入休息室

wait(doc); // 等待医生

signal(room); // 出休息室

signal(nwait); // 现在有一个乘客了

}

医生()

{

do {

wait(nwait);

print("治疗了一个病人");

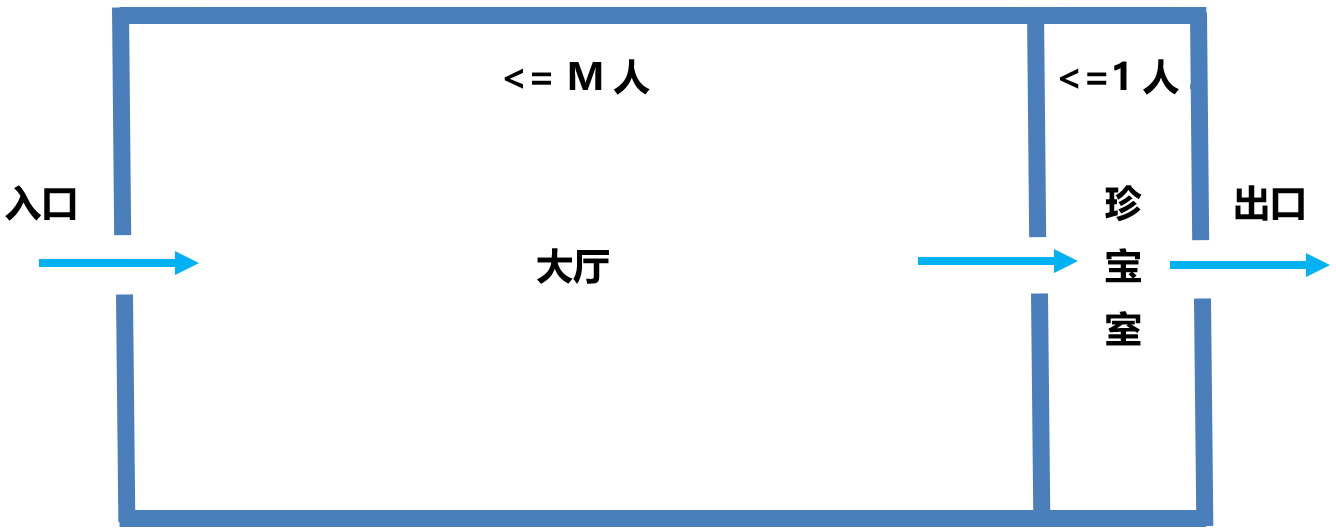
signal(doc);

} while(true);

}

**题目B：**

某小学组织同学们参观博物馆。博物馆由一个可供M个人参观的大厅，和仅供1个人参观的珍宝室组成。当大厅满员时，新来的参观者在大厅门外等候。珍宝室入口在大厅内，珍宝室出口就是博物馆出口。所有参观者均先在大厅参观，再进入珍宝室参观，最后从珍宝室出口离开博物馆。请用信号量及PV操作描述参观者的活动。



若为珍宝室内安排1名虚拟讲解员，其名字就叫“珍宝室”，并且显式地写出讲解员的活动进程，那么此题目就与题目A完全等效。只需把题目A中的相关信号量及过程名称改为本题目的即可。但这样写要看情况了，因为除了各个参观者进程外，再启动一个叫“珍宝室”的进程，也许不会被阅卷者认可。

下面是无虚拟讲解员(仅有参观者进程)的参考答案。

**答案B1：**

semaphore nHallEmpty = M; // 展览大厅空位数

semaphore mutexTreasure = 1; // 珍宝室互斥锁

visitor() // 每名参观者启动一个进程

{

P(nHallEmpty); // 进入大厅，或大厅满员时在大厅门外等待

visit\_the\_hall(); // 参观大厅

P(mutexTreasure); // 进入珍宝室，或珍宝室有人时在珍宝室门外等待

V(nHallEmpty); // 大厅空位+1，有可能唤醒大厅门外的等候者

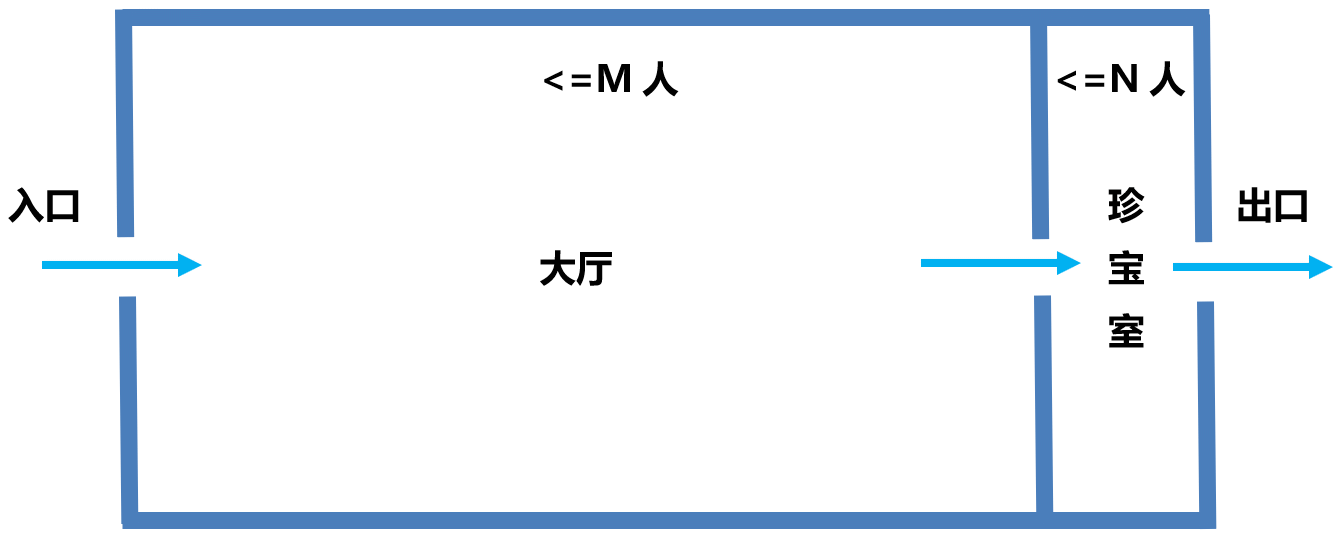
visit\_the\_treasure\_room(); // 参观珍宝室

V(mutexTreasure); // 离开珍宝室，释放珍宝室互斥锁

}

**题目C：**

某小学组织同学们参观博物馆。博物馆由一个可供M个人参观的大厅，和可供N个人参观的珍宝室组成。当大厅满员时，新来的参观者在大厅门外等候。珍宝室入口在大厅内，珍宝室出口就是博物馆出口。所有参观者均先在大厅参观，再进入珍宝室参观，最后从珍宝室出口离开博物馆。请用信号量及PV操作描述参观者的活动。



若为珍宝室内安排N名虚拟讲解员，其名字就叫“珍宝室”，并且显式地写出各个讲解员的活动进程，并为每一名讲解员启动一个进程，那么此题目就与后面的题目E完全等效。下面给出的是无虚拟讲解员(仅有参观者进程)的参考答案。

**答案C1：**

semaphore nHallEmpty = M; // 展览大厅空位数

semaphore nTreasureEmpty = N; // 珍宝室空位数

visitor() // 每名参观者启动一个进程

{

P(nHallEmpty); // 进入大厅，或大厅满员时在大厅门外等待

visit\_the\_hall(); // 参观大厅

P(nTreasureEmpty); // 进入珍宝室，或珍宝室满员时在珍宝室门外等待

V(nHallEmpty); // 大厅空位+1，有可能唤醒大厅门外的等候者

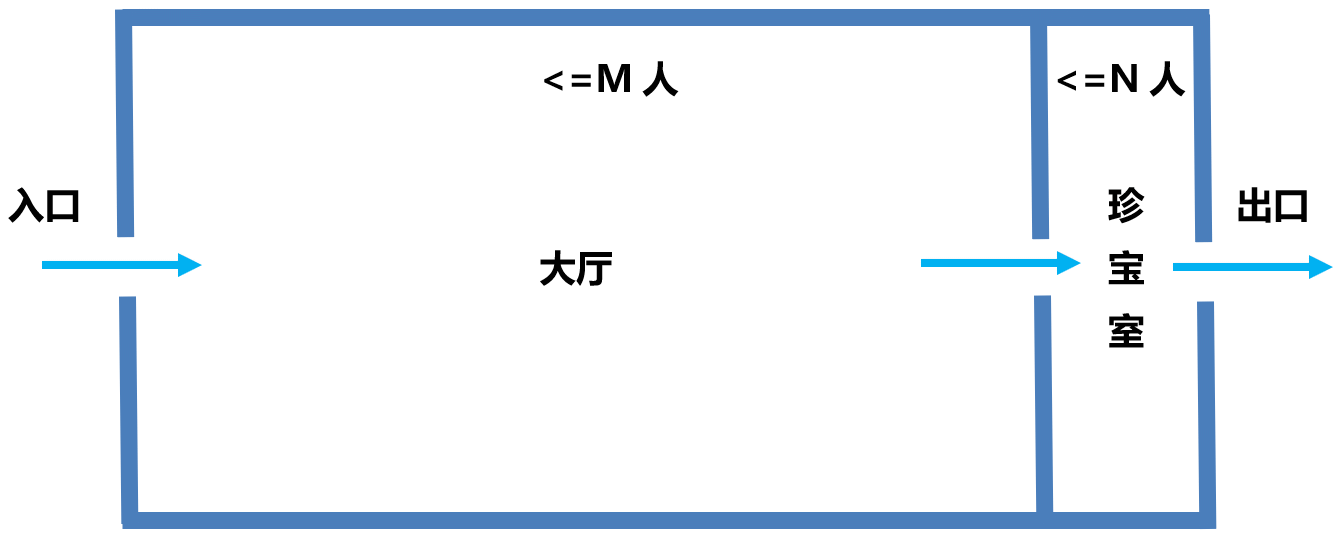
visit\_the\_treasure\_room(); // 参观珍宝室

V(nTreasureEmpty); // 离开珍宝室，有可能唤醒珍宝室门外的等候者

}

**题目D：**

某小学组织同学们参观博物馆。博物馆由一个可供M个人参观的大厅，和可供N个人参观的珍宝室组成。当大厅满员时，新来的参观者在大厅门外等候。珍宝室入口在大厅内，珍宝室出口就是博物馆出口。**珍宝室入口及出口仅容1人通过。**所有参观者均先在大厅参观，再进入珍宝室参观，最后从珍宝室出口离开博物馆。请用信号量及PV操作描述参观者的活动。



**答案D1：**

semaphore nHallEmpty = M; // 展览大厅空位数

semaphore nTreasureEmpty = N; // 珍宝室空位数

semaphore mutex\_entry = 1; // 珍宝室入口互斥锁

semaphore mutex\_exit = 1; // 珍宝室出口互斥锁

visitor() // 每名参观者启动一个进程

{

P(nHallEmpty); // 进入大厅，或大厅满员时在大厅门外等待

visit\_the\_hall(); // 参观大厅

P(nTreasureEmpty); // 等待珍宝室的空位

P(mutex\_entry); // 进入珍宝室入口

通过珍宝室入口();

V(mutex\_entry); // **S1A** 释放珍宝室入口

V(nHallEmpty); // **S1B** 大厅空位+1，有可能唤醒大厅门外的等候者

visit\_the\_treasure\_room(); // 参观珍宝室

P(mutex\_exit); // 进入珍宝室出口

通过珍宝室出口();

V(mutex\_exit); // **S2A** 离开珍宝室出口

V(nTreasureEmpty); // **S2B** 离开珍宝室，有可能唤醒珍宝室入口处的等候者

}

注：语句**S1A**与**S1B**可以交换顺序，语句**S2A**与**S2B**也可以交换顺序。

**题目E：**(题目及答案E1均来自前些天发的文档：**进程同步问题合集.docx**，答案是否正确请大家自行判断)

一医院门诊部的候诊室最多允许10名患者进入，若多于10人，则患者必须在门外等待，否则可进入候诊室候诊。门诊部共有3名医生，可同时给患者看病。当候诊室无等待的患者时，医生需等待患者的到来；当医生看病完成时，叫下一个等待的患者。当三个医生都在忙时，患者需等待，患者看病完成后离开候诊室。若把患者和医生看病的过程分别看作进程，试回答下列问题：（1）用信号量管理这些进程时，应怎样定义信号量，写出信号量的初值及信号量的含义；（2）根据所定义的信号量，用P、V原语描述患者进程和医生进程的活动，以保证它们能正确地并发执行。

**答案E1：**

解：（1）使用4个信号量实现，互斥信号量mutex保证病人互斥地进出候诊室。empty表示候诊室还可以容纳多少病人，full表示候诊室中有多少病人，service表示请求医生服务。

（2）同步算法描述如下：

Semaphore mutex=1；//互斥进出候诊室

Semaphore empty=10；//候诊室可进入人数

Semaphore full=0； //已进入候诊室人数

Semaphore service=0；//等待叫号

cobegin

{

process 病人i // 为每名病人各启动一个病人进程

{

P(empty)；

P(mutex)；

进入候诊室；

V(mutex)；

V(full)；

P(service)； // 等待叫号

获得服务；

服务结束;

P(mutex)；

走出候诊室；

V(mutex)；

V(empty);

}

process 医生i // 为每名医生各启动一个医生进程

{

while(TRUE)

{

P(full)；

V(service)；// 叫号

为病人诊病；

服务结束，病人离开；

}

}

}

coend

**答案E2：**

原题目并没有提到进出候诊室的门仅容一人通过，也没提到有病人取号和医生叫号过程。另外按照一般性常识，医生给病人看病时，并不在候诊区，而是在医生办公桌。本着最简原则，给出下面的答案。

semaphore empty=10；// 候诊室可进入人数

semaphore full=0； // 已进入候诊室人数

semaphore doctor=0； // 等待医生服务

process 病人i // 为每名病人各启动一个病人进程

{

P(empty)；

V(full)；

P(doctor)； // 等待医生

V(empty);

获得服务；

}

process 医生i // 为每名医生各启动一个医生进程

{

while(true)

{

P(full)； // 等待候诊室有病人，否则休息

V(doctor)； // 叫病人

为病人诊病；

}

}

**题目F：**(问题及答案均来自前些天发的文档：**部分OS试题.docx**)

本题为2013年研究生全国统考计算机试题操作系统部分题目之一

某博物馆最多可容纳500人同时参观，有一个出入口，该出入口一次仅允许一个人通过。参观者的活动描述如下。

|  |
| --- |
| cobegin       参观者进程 i：       {           …           进门；           …           参观；           …           出门；           …       }  coend |

请添加必要的信号量和 P、V（或wait()、signal()）操作，以实现上述过程中的互斥与同步。要求写出完整的过程，说明信号量的含义并赋初值。

答题要点：

// 定义两个信号量

Semaphore empty = 500;       / / 博物馆可以容纳的最多人数(2 分)

Semaphore mutex = 1;           / / 用于出入口资源的控制(2 分)

cobegin

参观者进程 i; // 每个到达的参观者启动一个进程

{

…

P ( empty );

P ( mutex );

进门;

V( mutex );

参观;

P ( mutex );

       出门;

V( mutex );

V( empty );

…

}

coend

**题目G：**(问题及答案均来自前些天发的文档：**部分OS试题.docx**)

本题为2011年研究生全国统考计算机试题操作系统部分题目之一

某银行提供1个服务窗口和10个供顾客等待的座位。顾客到达银行时，若有空座位，则到取号机上领取一个号，等待叫号。取号机每次仅允许一位顾客使用。当营业员空闲时，通过叫号选取一位顾客，并为其服务。顾客和营业员的活动过程描述如下。

|  |
| --- |
| cobegin       process  顾客 i       {           从取号机获取一个号码；           等待叫号；           获取服务；       }       process  营业员       {           while(TRUE) {                叫号；                为客户服务；           }       }  coend |

请添加必要的信号量和P、V(或 wait()、signal())操作，实现上述过程中的互斥与同步。要求写出完整的过程，说明信号量的含义并赋初值。

**答题要点：**

互斥资源：取号机(一次只允许一位顾客领号)，因此设一个互斥信号量mutex；

同步问题：顾客需要获得空座位等待叫号，当营业员空闲时，将选取一位顾客并为其服务。空座位的有、无影响等待顾客数量，顾客的有、无决定了营业员是否能开始服务，故分别设置信号量empty和full来实现这一同步关系。另外，顾客获得空座位后，需要等待叫号和被服务。这样，顾客与营业员就服务何时开始又构成了一个同步关系，定义信号量service来完成这一同步过程。

semaphore mutex=1；  // 互斥使用取号机

semaphore empty=10； // 空座位的数量

semaphore full=0；   // 已占座位的数量

semaphore service=0； // 等待叫号

cobegin

{

　　process顾客i

　　{

　　　　P(empty)；

　　　　P(mutex)；

　　　　从取号机获得一个号；

　　　　V(mutex)；

　　　　V(full)；

　　　　P(service)； // 等待叫号

　　　　获得服务；

}

　　process营业员

　　{

       　　while(TRUE)

　　　　{

              　 P(full)；

　　　　　　V(empty)；

　　　　　　V(service)；//叫号

　　　　　　为顾客服务；

　　　　}

　　}

}

coend

**进一步思考：**题目B-题目G改为与理发店问题一样，顾客(或乘客、病人)到达入口时，若休息室(或大厅等)没有空位置，选择离去，代码该如何编写？