1. **N个缓冲区的读写问题**

**是互斥+同步问题**

**进程：read（）、write（）**

semaphore mutex=1,full=0,empty=n;

void write(){

while(1){

得到一个结果;

p(empty); //消耗一个空闲缓冲区

p(mutex);

将数据写入缓冲区;

v(mutex);

v(full); //增加一个数据

}

}

void read(){

while(1){

p(full); //消耗一个数据;

p(mutex);

从缓冲区读一个数;

v(mutex);

v(empty); //增加一个空闲缓冲区;

将结果输出;

}

}

void main(){

cobegin

write();read();

coend

}

1. **对盘中可放N个水果的吃水果问题进行PV互斥同步**

**是互斥+同步问题**

**进程：dad（）、son（）、daughter（）**

semaphore mutex=1,S=N,So=0,Sa=0; //其中S指盘子里空出的可以放水果的个数

void dad(){

while(1){

p(S);

p(mutex);

将水果放入盘子;

v(mutex);

if(是橘子) v(So);

else v(Sa);

}

}

void son(){

while(1){

p(So);

p(mutex);

取橘子;

v(mutex);

v(S);

吃橘子;

}

}

void daughter(){

while(1){

p(Sa);

p(mutex);

取苹果;

v(mutex);

v(S);

吃苹果;

}

}

void main(){

cobegin

dad();son();daughter();

coend

}

1. **用管程解决读者写者问题**

**是互斥+同步问题**

(读者优先)

Monitor readwriter{

int count;

condition rmutex,wmutex; /\*其中rmutex用于保证对count的互斥访问，

wmutex用于实现对文件的互斥访问\*/

public:

void read1(){

cwait(rmutex); //各读进程互斥访问count

if(count==0)cwait(wmutex); //第一个读进程负责"上锁"

count++; //访问文件的读进程数+1

csignal(rmutex);

}

void read2(){

cwait(rmutex); //读进程互斥访问count

count--; //访问文件的读进程数-1

if(count==0)csignal(wmutex); //最后一个读进程负责"解锁"

csignal(rmutex);

}

void write1(){

cwait(wmutex); //写之前"上锁"

}

void write2(){

csignal(wmutex); //写之后"解锁"

}

{count=0;}

}RW;

void reader(){

while(1){

RW.read1();

read something;

RW.read2();

}

}

void writer(){

while(1){

RW.write1();

write something;

RW.write2();

}

}

void main(){

cobegin

reader();writer();

coend

}

(写者优先)

Monitor readwriter{

int count;

condition rmutex,wmutex; /\*其中rmutex用于保证对count的互斥访问，

wmutex用于实现对文件的互斥访问\*/

condition w; //用于实现"写优先"

public:

void read1(){

cwait(w);

cwait(rmutex); //各读进程互斥访问count

if(count==0)cwait(wmutex); //第一个读进程负责"上锁"

count++; //访问文件的读进程数+1

csignal(rmutex);

csignal(w);

}

void read2(){

cwait(rmutex); //读进程互斥访问count

count--; //访问文件的读进程数-1

if(count==0)csignal(wmutex); //最后一个读进程负责"解锁"

csignal(rmutex);

}

void write1(){

cwait(w);

cwait(wmutex); //写之前"上锁"

}

void write2(){

csignal(wmutex); //写之后"解锁"

csignal(w);

}

{count=0;}

}RW;

void reader(){

while(1){

RW.read1();

read something;

RW.read2();

}

}

void writer(){

while(1){

RW.write1();

write something;

RW.write2();

}

}

void main(){

cobegin

reader();writer();

coend

}