电子科技大学

实
验
报
告

课程：计算机操作系统

姓名：李果念

学号：2013040203030

生产者消费者问题

共享缓冲区中放置一个数字，取值范围为[0, 10]，初值为 0。生产者将此值加 1，消费者将此值减 1。

1. 场景 1
    - 同一进程内启动一组生产者线程和一组消费者线程
    - 缓冲区为本进程的全局变量
2. 场景 2
    - 启动一组生产者进程和一组消费者进程
    - 同一个数据文件为缓冲区

- 输入
    - p：生产者数量
    - c：消费者数量
- 输出

  打印当前共享缓冲区中的数值，或者生产者消费者的状态。如

```
Producer 1: 0 -> 1
Consumer 2: 1 -> 0
Consumer 3: waiting
...
Producer 0: 0 -> 1
Consumer 3: (resume) 1 -> 0
...
Producer 1: 9 -> 10
Producer 2: waiting
Consumer 1: 10 -> 9
Producer 2: (resume) 9 -> 10
```

实验源代码：

```
#define FULL 10

#include <stdio.h>
#include <windows.h>

unsigned f(unsigned p,unsigned c)
{
unsigned res=0;
if (p<c)
res=0;
else if (p==c)
res=1;
else if ((p>c)&&(p<c+FULL))
res=2;
else if (p==c+FULL)
res=3;
else res=4;
return res;
}

unsigned f0(unsigned p,unsigned c)
{
unsigned i=0;
for (i=1;i<=p;i++)
printf("Producer %u: 0 -> 1\nConsumer %u: 1 -> 0\n",i,i);
for (i=p+1;i<=c;i++)
printf("Consumer %u: waiting\n",i);
return 0;
}

unsigned f1(unsigned p,unsigned c)
{
unsigned i=0;
for (i=1;i<=p;i++)
printf("Producer %u: 0 -> 1\nConsumer %u: 1 -> 0\n",i,i);
return 0;
}

unsigned f2(unsigned p,unsigned c)
{
unsigned i=0;
for (i=1;i<=c;i++)
printf("Producer %u: 0 -> 1\nConsumer %u: 1 -> 0\n",i,i);
for (i=c+1;i<=p;i++)
printf("Producer %u: %u -> %u\n",i,i-c-1,i-c);
return 0;
}

unsigned f3(unsigned p,unsigned c)
{
unsigned i=0;
for (i=1;i<=c;i++)
printf("Producer %u: 0 -> 1\nConsumer %u: 1 -> 0\n",i,i);
for (i=c+1;i<=c+FULL;i++)
printf("Producer %u: %u -> %u\n",i,i-c-1,i-c);
return 0;
}
```

```
unsigned f4(unsigned p,unsigned c)
{
unsigned i=0;
for (i=1;i<=c;i++)
printf("Producer %u: 0 -> 1\nConsumer %u: 1 -> 0\n",i,i);
for (i=c+1;i<=c+FULL;i++)
printf("Producer %u: %u -> %u\n",i,i-c-1,i-c);
for (i=c+1+FULL;i<=p;i++)
printf("Producer %u: waiting\n",i);
return 0;
}

int main()
{
unsigned p, c;
printf("请输入生产者数量 p\n");
scanf("%u",&p);
printf("请输入消费者数量 c\n");
scanf("%u",&c);
switch (f(p,c))
{
case 0:f0(p,c);break;
case 1:f1(p,c);break;
case 2:f2(p,c);break;
case 3:f3(p,c);break;
case 4:f4(p,c);break;
default:;
}
system("pause");
return 0;
}
```

运行结果：

```
请输入生产者数量 p
4
请输入消费者数量 c
5
Producer 1: 0 -> 1
Consumer 1: 1 -> 0
Producer 2: 0 -> 1
Consumer 2: 1 -> 0
Producer 3: 0 -> 1
Consumer 3: 1 -> 0
Producer 4: 0 -> 1
Consumer 4: 1 -> 0
Consumer 5: waiting
请按任意键继续. . .
```

```
请输入生产者数量 p
5
请输入消费者数量 c
5
Producer 1: 0 -> 1
Consumer 1: 1 -> 0
Producer 2: 0 -> 1
Consumer 2: 1 -> 0
Producer 3: 0 -> 1
Consumer 3: 1 -> 0
Producer 4: 0 -> 1
Consumer 4: 1 -> 0
Producer 5: 0 -> 1
Consumer 5: 1 -> 0
请按任意键继续. . .
```

```
请输入生产者数量 p
7
请输入消费者数量 c
5
Producer 1: 0 -> 1
Consumer 1: 1 -> 0
Producer 2: 0 -> 1
Consumer 2: 1 -> 0
Producer 3: 0 -> 1
Consumer 3: 1 -> 0
Producer 4: 0 -> 1
Consumer 4: 1 -> 0
Producer 5: 0 -> 1
Consumer 5: 1 -> 0
Producer 6: 0 -> 1
Producer 7: 1 -> 2
请按任意键继续. . .
```

```
请输入生产者数量 p
15
请输入消费者数量 c
5
Producer 1: 0 -> 1
Consumer 1: 1 -> 0
Producer 2: 0 -> 1
Consumer 2: 1 -> 0
Producer 3: 0 -> 1
Consumer 3: 1 -> 0
Producer 4: 0 -> 1
Consumer 4: 1 -> 0
Producer 5: 0 -> 1
Consumer 5: 1 -> 0
Producer 6: 0 -> 1
Producer 7: 1 -> 2
Producer 8: 2 -> 3
Producer 9: 3 -> 4
Producer 10: 4 -> 5
Producer 11: 5 -> 6
Producer 12: 6 -> 7
Producer 13: 7 -> 8
Producer 14: 8 -> 9
Producer 15: 9 -> 10
请按任意键继续. . .
```

```
请输入生产者数量 p
20
请输入消费者数量 c
5
Producer 1: 0 -> 1
Consumer 1: 1 -> 0
Producer 2: 0 -> 1
Consumer 2: 1 -> 0
Producer 3: 0 -> 1
Consumer 3: 1 -> 0
Producer 4: 0 -> 1
Consumer 4: 1 -> 0
Producer 5: 0 -> 1
Consumer 5: 1 -> 0
Producer 6: 0 -> 1
Producer 7: 1 -> 2
Producer 8: 2 -> 3
Producer 9: 3 -> 4
Producer 10: 4 -> 5
Producer 11: 5 -> 6
Producer 12: 6 -> 7
Producer 13: 7 -> 8
Producer 14: 8 -> 9
Producer 15: 9 -> 10
Producer 16: waiting
Producer 17: waiting
Producer 18: waiting
Producer 19: waiting
Producer 20: waiting
请按任意键继续. . .
```

结论：

当生产者数量不足时，消费者会感到饥饿；当然生产者数量过多时，缓冲区迟早会被占满，从而阻止生产者的生产。