生产者消费者问题

当缓冲区为空时，in 与 out 在数值上相等；当缓冲区满时，in 与 out 在数值上也相等。那么，这就产生了不能判断的情况，不能通过 in 与 out 在数值上是否相等来判断缓冲区是否为空或者是否为满，本质上是循环队列产生的问题。这种问题有很多解决方案，除了牺牲一个位置以外，增加一个标识、增加计数器或者用特殊值来表示等等都是可以的。

- 方案一：增加计数器 count

1、操作

count 初始值为 0；

当生产者向缓冲区增加一项时，count 自增 1；

当消费者从缓冲区移走一项时，count 自减 1

2、代码

1）生产者

```
while (true){
    while (count == BUFFER_SIZE)
        ; /* do nothing -- no free buffers */
    buffer[in] = nextProduced;
    in = (in + 1) % BUFFER_SIZE;
    count ++;
}
```

2）消费者

```
while (true){
    while(count == 0)
        ; /*do nothing -- nothing to consume */
    nextConsumed = buffer[out];
    out = (out + 1) % BUFFER_SIZE;
    count --;
}
```

3、问题

当生产者和消费展的代码并发执行时可能不能正确运行，这是由于“count++”和“count--”在机器码上是通过多条语句实现的（需要借助寄存器）。那么，执行可能会按照任意顺序交替执行，从而导致不同步的问题。

count++的机器语言：	count--的机器语言：
register1 = count	register2 = count
register1 = register1 + 1	register2 = register2 - 1
count = register1	count = register

T_0: producer execute $register_1$ = counter { $register_1$ = 5}

T_1: producer execute $register_1$ = $register_1$+1 { $register_1$ = 6}

T_2: consumer execute $register_2$ = counter { $register_2$ = 5}

T_3: consumer execute $register_2$ = $register_2$−1 { $register_2$ = 4}

T_4: producer execute counter = $register_1$ { $register_1$ = 6}

T_5: consumer execute counter = $register_2$ { $register_2$ = 4}

- 方案二：利用 flag 作为标识

1、操作

使用布尔类型的变量 full 来表示缓冲区是否已满，当值为 true 表示缓冲区已满，当值为 false 表示缓冲区未满；

初始值为 false，在生产者向缓冲区中增加项时进行检查

2、代码

初始化：bool full = false;

1）生产者

```
while (true){
    while (full)
        ; /* do nothing -- no free buffers */
    buffer[in] = nextProduced;
    in = (in + 1) % BUFFER_SIZE;
    if (in == out){
        full = true;
    }
}
```

2）消费者

```
while (true){
    while(in == out && !full)
        ; /*do nothing -- nothing to consume */
    nextConsumed = buffer[out];
    out = (out + 1) % BUFFER_SIZE;
    if (full){
        full = false;
    }
}
```