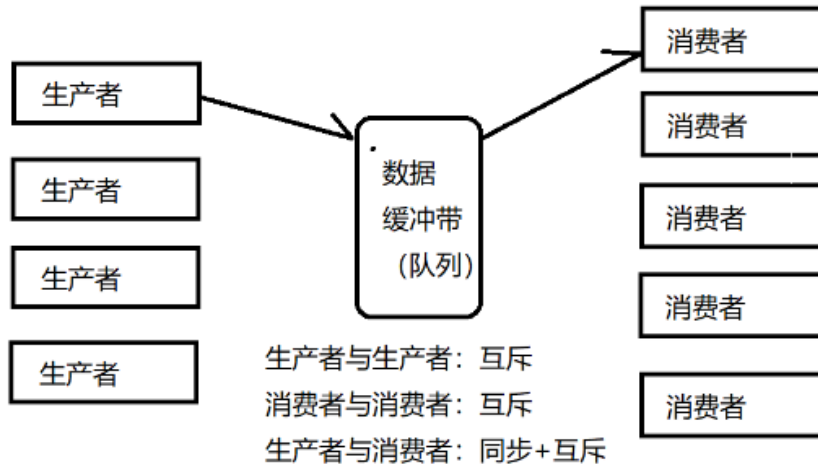


生产者与消费者模型:一种设计模式---大佬们针对典型场景设计的解决方案



生产者与消费者模型所解决的问题: 解耦和, 支持并发, 支持忙闲不均

生产者与消费者模型的实现:

生产者与消费者只是不同角色的执行流;

只需要中间实现线程安全的队列, 然后再各自创建不同角色的执行流就可以实现这个模型

线程安全的阻塞队列的实现:

使用c++封装实现一个阻塞队列类:: stl中我们学习了队列容器std::queue--但是这个队列是非线程安全的

```
class BlockQueue{
private:
    std::queue<int> _queue; //stl中queue
    int _capacity; //定义队列中最大的节点数量
    pthread_mutex_t _mutex; //互斥锁保护临界资源queue的访问
    pthread_cond_t _cond_con; //消费者等待队列, 实现同步
    pthread_cond_t _cond_pro; //生产者等待队列, 实现同步
public:
    BlockQueue(int max_que = MAX_QUEE).初始化过程..}
    ~ BlockQueue () {..资源销毁过程..}
    bool Push(int data); //生产者入队操作
    bool Pop(int *data); //消费者出队操作
}
```

生产者与消费者模型:

- 1.这个模型的应用场景:
- 2.这个模型针对这种场景解决了什么问题:
- 3.生产者与消费者模型的实现流程

POSIX标准信号量:千万不要跟信号概念混淆了

作用:实现进程/线程间的同步与互斥

本质:一个计数器+ pcb等待队列

实现同步:

计数器对资源数量进行计数, 当线程想要获取资源的时候, 先访问信号量,判断是否能够获取(信号量通过自身的计数完成判断),若计数 <= 0则直接阻塞线程, 计数-1;其它线程生产资源之后, 计数+1,唤醒等待队列上的pcb

实现互斥:

保证计数器的数值不会大于1,就表示同一时间只有一个线程能够访问资源;

操作接口:

1.定义信号量sem_t sem;

2.初始化信号量int sem_init(sem_t *sem, int pshared, int value)

pshared:信号量即可用于进程间也可用于线程间, pshared为0表示用于线程间 非0表示用于进程间

value:信号量就是一个计数器, 统计资源数量, value就是通过资源数量初始化计数器的

3.在访问临界资源之前, 先判断计数, 是否能够访问资源, 若不能访问, 则阻塞线程;若可以访问则调用直接返回

int sem_wait(sem_t *sem) / int sem_trywait(sem_t *sem)/ int sem_timedwait(sem_t *sem, struct timespec *ts)

4.访问临界资源之后/生产资源之后, 唤醒一个等待的线程, 并且计数+ 1

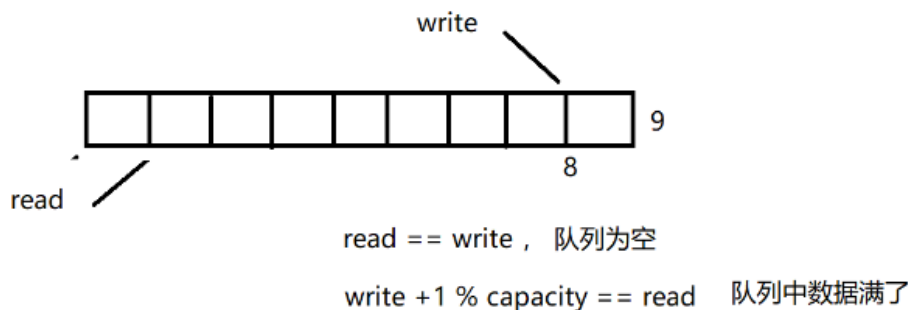
int sem_post(sem_t *sem);

5.不使用信号量记得释放资源

int sem_destroy(sem_t *sem);

信号量的应用:

通过信号量实现一个环形队列, 最终实现一个生产者与消费者模型



```
class RingQueue{
```

```
private:
```

```
    std::vector <int> _queue(node_num);
```

```
    int _step_read; //当前即将读取数据的位置的下标
```

```
    int _step_write; //当前即将写入数据的位置的下标
```

```
    sem_t _sem_lock; //用于实现互斥的信号量
```

```
    sem_t _sem_data; //使用这个计数器,实现对当前队列中的数据资源的数量进行计数;如果 <= 0 表示没有资源, 则消费者会陷入等待
```

```
    sem_t _sem_space; //使用这个计数器, 实现对当前队列中的空闲空间数量进行计数;如果 <= 0 表示队列满了, 则生产者陷入等地啊
```

```
public:
```

```
    RingQueue (int max = MAX_QUEUE) {...初始化过程.}
```

```
    ~ RingQueue(){...销毁资源过程..}
```

```
    bool Push(int data);
```

```
    bool Pop(int *data);
```

```
}
```

信号量:

1.信号量的本质

2.信号量的作用

3.信号量实现互斥的原理

4.信号量实现同步的原理

5.信号量实现环形队列的流程实现

