

# Mutual Exclusion

We want to use **mutual exclusion** to synchronize access to shared resources

Code that uses mutual exclusion to synchronize its execution is called a **critical section**

- Only one thread at a time can execute in the critical section
- All other threads are forced to wait on entry
- When a thread leaves a critical section, another can enter

# A classical example - Producer Consumer

```
void producer () {  
    while(1) {  
        item := produce()  
        while(full(buffer)) {  
            /* do nothing */  
        }  
        write(buffer, item)  
    }  
}
```

```
void consumer () {  
    while(1) {  
        while(empty(buffer)) {  
            /* do nothing */  
        }  
        item := read(buffer)  
        consume(item)  
    }  
}
```