# 程序说明

### 读者优先

读者优先和写者优先需要引入一个状态变量state，表明当前系统的状态。需要用到的几个状态量如下：

* mutex：保证reader*count, writer*count, state变量访问的互斥性
* Sig\_read：允许读的信号量
* Sig\_wrt：允许写的信号量

#### 读者

wait(mutex);  
++reader\_count;  
if(state == s\_waiting || state == s\_reading) {  
 signal(Sig\_read);  
 state = s\_reading;  
}  
signal(mutex);  
  
// read  
  
wait(mutex);  
--reader\_count;  
if(reader\_count == 0) {  
 if(writer\_count != 0) {  
 signal(Sig\_wrt);  
 state = s\_writing;  
 } else {  
 state = s\_waiting;  
 }  
}  
signal(mutex);

#### 写者

wait(mutex);  
++writer\_count;  
if(state == s\_waiting) {  
 signal(Sig\_wrt);  
 state = s\_writing;  
}  
signal(mutex);  
  
// write  
  
wait(mutex);  
--writer\_count;  
if(reader\_count != 0) {  
 signal(Sig\_read);  
 state = s\_reading;  
} else if(writer\_count != 0) {  
 signal(Sig\_wrt);  
 state = s\_writing;  
} else {  
 state = s\_waiting;  
}  
signal(mutex);

### 写者优先

#### 读者

wait(mutex);  
++reader\_count;  
if(state == s\_waiting   
|| (state == s\_reading && writer\_count == 0)) {  
 signal(Sig\_read);  
 state = s\_reading;  
}  
signal(mutex);  
  
// read  
  
wait(mutex);  
--reader\_count;  
if(writer\_count != 0) {  
 // 有读者在等待  
 sem\_post(Sig\_wrt);  
 state = s\_writing;  
} else if(reader\_count == 0) {  
 // 等待队列为空  
 state = s\_waiting;  
}  
signal(mutex);

#### 写者

wait(mutex);  
++writer\_count;  
if(state == s\_waiting) {  
 signal(Sig\_wrt);  
 state = s\_writing;  
}  
signal(mutex);  
  
// write  
  
wait(mutex);  
--writer\_count;  
if(writer\_count != 0) {  
 signal(Sig\_wrt);  
 state = s\_writing;  
} else if(reader\_count != 0) {  
 for(int i=0; i!=reader\_count; ++i) {  
 signal(Sig\_read);  
 }  
 state = s\_reading;  
} else {  
 state = s\_waiting;  
}  
signal(mutex);