Operating Systems and Concurrency

Lecture 11: Concurrency + Revision

G52OSC/COMP2007

Geert De Maere
(Isaac Triguero)
{Geert.DeMaere,Isaac.Triguero}@Nottingham.ac.uk

University Of Nottingham
United Kingdom

2018

Assignment Project Exam Help

https://powcoder.com

- Writers before readers
- Past exam questions

Add WeChat powcoder

Assignment Project Exam Help

- Solution 2 (readers before writers): unless `iReadCount` reaches 0, **writing will not happen**
  - The writer can easily **starve** if there are **multiple readers**

https://powcoder.com

- Solution 3 gives priority to **writers** and uses:
  - `iReadCount`, `iWriteCount` to keep track of the number of readers/writers
  - `sRead`/`sWrite` to synchronise the **reader's/writer's critical section**

Add WeChat powcoder

  - `sReadTry` to stop readers when there is **a writer waiting**
  - `sResource` to **synchronise** the resource for **reading/writing**

```c
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```c
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```c
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry); // 1=>0
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```c
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```c
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead); // 1=>0
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```c
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead);
    iReadCount++;// 0=>1
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```
void * reader(void * arg)              void * writer(void * arg)
{                                      {
  while(1)                               while(1)
  {                                      {
    sem_wait(&sReadTry);                   sem_wait(&sWrite);
    sem_wait(&sRead);                      iWriteCount++;
    iReadCount++;                          if(iWriteCount == 1)
    if(iReadCount == 1)                      sem_wait(&sReadTry);
      sem_wait(&sResource); //I>0         sem_post(&sWrite);
    sem_post(&sRead);
    sem_post(&sReadTry);                   sem_wait(&sResource);
                                           printf("writing\n");
    printf("reading\n");                   sem_post(&sResource);

    sem_wait(&sRead);                      sem_wait(&sWrite);
    iReadCount--;                          iWriteCount--;
    if(iReadCount == 0)                    if(iWriteCount == 0)
      sem_post(&sResource);                  sem_post(&sReadTry);
    sem_post(&sRead);                      sem_post(&sWrite);
  }                                      }
}                                      }
```

```
void * reader(void * arg)              void * writer(void * arg)
{                                      {
  while(1)                               while(1)
  {                                      {
    sem_wait(&sReadTry);                   sem_wait(&sWrite);
    sem_wait(&sRead);                      iWriteCount++;
    iReadCount++;                          if(iWriteCount == 1)
    if(iReadCount == 1)                      sem_wait(&sReadTry);
      sem_wait(&sResource);              sem_post(&sWrite);
    sem_post(&sRead); // 0=>1
    sem_post(&sReadTry);                   sem_wait(&sResource);
                                           printf("writing\n");
    printf("reading\n");                   sem_post(&sResource);

    sem_wait(&sRead);                      sem_wait(&sWrite);
    iReadCount--;                          iWriteCount--;
    if(iReadCount == 0)                    if(iWriteCount == 0)
      sem_post(&sResource);                  sem_post(&sReadTry);
    sem_post(&sRead);                      sem_post(&sWrite);
  }                                      }
}                                      }
```

```c
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);// 0=>1

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```c
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite); // 1=>0
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

```
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++; // 0=>1
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry); // 1=>0
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```
void * reader(void * arg)          void * writer(void * arg)
{                                  {
  while(1)                           while(1)
  {                                  {
    sem_wait(&sReadTry);               sem_wait(&sWrite);
    sem_wait(&sRead);                  iWriteCount++;
    iReadCount++;                      if(iWriteCount == 1)
    if(iReadCount == 1)                  sem_wait(&sReadTry);
      sem_wait(&sResource);            sem_post(&sWrite);   // w=>1
    sem_post(&sRead);
    sem_post(&sReadTry);               sem_wait(&sResource);
                                       printf("writing\n");
    printf("reading\n");               sem_post(&sResource);

    sem_wait(&sRead);                  sem_wait(&sWrite);
    iReadCount--;                      iWriteCount--;
    if(iReadCount == 0)                if(iWriteCount == 0)
      sem_post(&sResource);              sem_post(&sReadTry);
    sem_post(&sRead);                  sem_post(&sWrite);
  }                                  }
}                                  }
```

```
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource); // 1->0
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource); // 0=>-1 (sleep)
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```
void * reader(void * arg)           void * writer(void * arg)
{                                   {
  while(1)                            while(1)
  {                                   {
    sem_wait(&sReadTry);                sem_wait(&sWrite);
    sem_wait(&sRead);                   iWriteCount++;
    iReadCount++;                       if(iWriteCount == 1)
    if(iReadCount == 1)                   sem_wait(&sReadTry);
      sem_wait(&sResource);             sem_post(&sWrite);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");               sem_wait(&sResource);
                                       printf("writing\n");
                                       sem_post(&sResource);
    sem_wait(&sRead);
    iReadCount--;                       sem_wait(&sWrite);
    if(iReadCount == 0)                 iWriteCount--;
      sem_post(&sResource);             if(iWriteCount == 0)
    sem_post(&sRead);                     sem_post(&sReadTry);
  }                                     sem_post(&sWrite);
}                                     }
                                    }
```

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

```
void * reader(void * arg)          void * writer(void * arg)
{                                  {
  while(1)                           while(1)
  {                                  {
    sem_wait(&sReadTry);               sem_wait(&sWrite);
    sem_wait(&sRead);                  iWriteCount++;
    iReadCount++;                      if(iWriteCount == 1)
    if(iReadCount == 1)                  sem_wait(&sReadTry);
      sem_wait(&sResource);            sem_post(&sWrite);
    sem_post(&sRead);
    sem_post(&sReadTry);
                                       sem_wait(&sResource);
                                       printf("writing\n");
    printf("reading\n");               sem_post(&sResource);

    sem_wait(&sRead); // 1>>           sem_wait(&sWrite);
    iReadCount--;                      iWriteCount--;
    if(iReadCount == 0)                if(iWriteCount == 0)
      sem_post(&sResource);              sem_post(&sReadTry);
    sem_post(&sRead);                  sem_post(&sWrite);
  }                                  }
}                                  }
```

```
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--; // 1=>0
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```
void * reader(void * arg)              void * writer(void * arg)
{                                      {
  while(1)                               while(1)
  {                                      {
    sem_wait(&sReadTry);                   sem_wait(&sWrite);
    sem_wait(&sRead);                      iWriteCount++;
    iReadCount++;                          if(iWriteCount == 1)
    if(iReadCount == 1)                      sem_wait(&sReadTry);
      sem_wait(&sResource);              sem_post(&sWrite);
    sem_post(&sRead);
    sem_post(&sReadTry);                   sem_wait(&sResource); // wakeup
                                           printf("writing\n");
    printf("reading\n");                   sem_post(&sResource);

    sem_wait(&sRead);                      sem_wait(&sWrite);
    iReadCount--;                          iWriteCount--;
    if(iReadCount == 0)                    if(iWriteCount == 0)
      sem_post(&sResource); // -1=>0         sem_post(&sReadTry);
    sem_post(&sRead);                      sem_post(&sWrite);
  }                                      }
}                                      }
```

```c
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead); // 0=>1
  }
}
```

```c
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry); // 0=>-1
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry); // 1=>0
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```c
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```c
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource); // (woken up)
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```
void * reader(void * arg)               void * writer(void * arg)
{                                        {
  while(1)                                 while(1)
  {                                        {
    sem_wait(&sReadTry);                     sem_wait(&sWrite);
    sem_wait(&sRead);                        iWriteCount++;
    iReadCount++;                            if(iWriteCount == 1)
    if(iReadCount == 1)                        sem_wait(&sReadTry);
      sem_wait(&sResource);                  sem_post(&sWrite);
    sem_post(&sRead);
    sem_post(&sReadTry);
                                             sem_wait(&sResource);
                                             printf("writing\n");
    printf("reading\n");                     sem_post(&sResource);

    sem_wait(&sRead);                        sem_wait(&sWrite);
    iReadCount--;                            iWriteCount--;
    if(iReadCount == 0)                      if(iWriteCount == 0)
      sem_post(&sResource);                    sem_post(&sReadTry);
    sem_post(&sRead);                        sem_post(&sWrite);
  }                                        }
}                                        }
```

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

```
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource); // 0=>1

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--; // 1=>0
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite);
  }
}
```

```c
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);  // wakeup
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```c
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry); // -1=>0
    sem_post(&sWrite);
  }
}
```

```
void * reader(void * arg)
{
  while(1)
  {
    sem_wait(&sReadTry);
    sem_wait(&sRead);
    iReadCount++;
    if(iReadCount == 1)
      sem_wait(&sResource);
    sem_post(&sRead);
    sem_post(&sReadTry);

    printf("reading\n");

    sem_wait(&sRead);
    iReadCount--;
    if(iReadCount == 0)
      sem_post(&sResource);
    sem_post(&sRead);
  }
}
```

```
void * writer(void * arg)
{
  while(1)
  {
    sem_wait(&sWrite);
    iWriteCount++;
    if(iWriteCount == 1)
      sem_wait(&sReadTry);
    sem_post(&sWrite);

    sem_wait(&sResource);
    printf("writing\n");
    sem_post(&sResource);

    sem_wait(&sWrite);
    iWriteCount--;
    if(iWriteCount == 0)
      sem_post(&sReadTry);
    sem_post(&sWrite); // 0=>1
  }
}
```

Assignment Project Exam Help

https://powcoder.com

- Readers/writers problem
  - Solution with **minimum parallelism**
  - Solution with **priority for the readers**
  - Solution with **priority for the writers**

Add WeChat powcoder

Question 1
Comprehension

Assignment Project Exam Help

"Describe how, in your opinion, **recent developments in computer architecture** and computer design have **influenced operating system design**. Include aspects on **processes management**, **memory management**, **file system design**, and **disks**. " [10 marks]

https://powcoder.com

Add WeChat powcoder

Question 2
Comprehension

Assignment Project Exam Help

https://powcoder.com

*"Keeping in mind the implementation differences between threads and processes on an operating system level, explain **why thread creation and switching** are **faster than process creation and switching**." [3 marks]*

Add WeChat powcoder

Question 2
Comprehension

Assignment Project Exam Help

https://powcoder.com

*"In which situations would you favour **user level threads**? In which situations would you favour **kernel level threads**. Explain your answer."* [2 marks]

Add WeChat powcoder

Question 2
Comprehension

Assignment Project Exam Help

https://powcoder.com

"**Round Robin** is said to favour **CPU bound processes** over I/O bound processes. Explain why this may be the case (if this is the case at all) [2 marks]? "

Add WeChat powcoder

Question 2
Application

"Illustrate the use of **round robin**, **shortest job first**, and **priority queues** for the processes given below (all processes arrive at the same time)."

- Give the **order** in which the processes are scheduled, together with the times at which they will start, end, are interrupted. You can assume a **time slice of 15 milliseconds** for the round robin algorithm. You can illustrate your answer using Gantt charts if you prefer [3 marks].

- What is the **average response time** for each of the algorithms above [3 marks]?

- What is the **average turnaround time** for each of the algorithms above [3 marks]?

Question 2
Application

	FCFS Position	CPU burst time	Priority
Process A	1	31	1 (high)
Process B	2	57	1 (high)
Process C	3	11	2 (low)
Process D	4	15	2 (low)

Question 4
Knowledge

Assignment Project Exam Help

https://powcoder.com

"Describe the principle behind address relocation. Why is it necessary and why is it useful?" [4 marks]

Add WeChat powcoder

Question 4
Comprehension

Assignment Project Exam Help

"*Out of the following four scheduling algorithms, which one(s) can lead to starvation: **FCFS**, **shortest job first**, **round robin**, **highest priority first**? Explain your answer.*" [2 marks]

https://powcoder.com

Add WeChat powcoder

Recap
Take-Home Message

Assignment Project Exam Help

- Writers before Readers
https://powcoder.com
- Exam Questions
- Next lecture: Memory Management (Isaac Triguero)

Add WeChat powcoder