**为了方便检查，请大家使用统一的输入输出格式**

**理发师问题**

**输入格式：**

**输入的第1行为4个整数，分别代表：顾客数n、理发师数、椅子数、理发时间（秒）；第2~n+1行输入n个顾客的情况，每行输入2个变量，输入的数据依次为：第i个顾客、顾客的到达时间。**

**输出格式：**

**顾客进店时，如果有机会理发（有座位或能够直接开始理发），打印输出：**

**“customer i: there are n customers in front waiting for a haircut”**

**如果剩余椅子数为0，顾客直接离开，打印输出：**

**“customer i: no more empty chairs, customer leaves”**

**顾客理发完毕时，打印输出：**

**“customer i finished haircut”**

**例如：**

|  |  |
| --- | --- |
| **Customer**  Down(mutex);  if(emptyChairNum > 0){  **printf(“customer %d: there are n %d customers in front waiting for a haircut”, id, totalNum);**  custOnChair ++;  Up(wait);  Up(mutex);  Down(barber);  getHaircut  **printf(“customer %d finished haircut, waiting line num: %d”, id);**  }  else{  **printf(“customer %d: no more empty chairs, customer leaves**”**, id)**;  Up(mutex);  } | **Barber**  Down(wait);  Down(mutex);  Up(barber);  custOnChair --;  Up(mutex);  Haircut |

**示例输入输出：**

|  |  |
| --- | --- |
| **输入** | **输出** |
| **5 1 1 3**  **1 7**  **2 3**  **3 2**  **4 4**  **5 10** | **customer 3: there are 0 customers in front waiting for a haircut**  **customer 2: there are 0 customers in front waiting for a haircut**  **customer 4: no more empty chairs, customer leaves**  **customer 3 finished haircut**  **customer 1: there are 0 customers in front waiting for a haircut**  **customer 2 finished haircut**  **customer 5: there are 0 customers in front waiting for a haircut**  **customer 1 finished haircut**  **customer 5 finished haircut** |

* **注意: 在输入完成后再统一创建线程**

**读写者问题**

**输入格式：**

输入的第一行为两个整数，第一个整数表明读写者的优先顺序（1代表读者优先，2代表写者优先），第二个整数表明接下来将要输入的读写者数量。接下来每行输入4个变量，输入的数据依次为：进程编号i、读写者、进程i开始执行的时间、进程i持续时间。最后一行键入回车代表结束。

**输出格式：**

每次读写者申请读写时，输出: “writer(/reader) i waiting to write(/read)”;

当读写者开始读写时，输出：“writer(/reader) i starts to write(/read)”;

当读写者完成读写时，输出：“writer(/reader) i ends writing(/read)”

**例如，**在读者优先的读写者问题中，示例输出打印如下：

|  |  |
| --- | --- |
| **Reader**  Down(mutex);  **printf(“reader %d waiting to read”, n);**  reader++;  if(reader == 1) Down(wmutex);  Up(mutex);  **printf(“reader %d starts to read”, n);**  Read data  **printf(“reader %d ends reading”, n);**  Down(mutex);  reader--;  if(reader == 0) Up(wmutex);  Up(mutex); | **Writer**  **printf(“writer %d waiting to write”, n);**  Down(wmutex);  **printf(“writer %d starts to write”, n);**  Write data  **printf(“writer %d ends writing”, n);**  Up(wmutex); |

**输入输出样例：**

|  |  |
| --- | --- |
| **输入** | **输出** |
| 1 5  1 R 3 5  2 W 4 5  3 R 5 2  4 R 6 5  5 W 9 3 | reader 1 waiting to read  reader 1 starts to read  writer 2 waiting to write  reader 3 waiting to read  reader 3 starts to read  reader 4 waiting to read  reader 4 starts to read  reader 3 ends reading  reader 1 ends reading  writer 5 waiting to write  reader 4 ends reading  writer 2 starts to write  writer 2 ends writing  writer 5 starts to write  writer 5 ends writing |