Multiple Readers/Consecutive Writers Monitor

```
int reading = 0;          // Number of readers reading (0 or more)
int readers = 0;          // Number of readers reading or waiting to read
int writing = 0;          // Number of writers writing (0 or 1)
int writers = 0;          // Number of writers writing or waiting to write
                          // excluding writers waiting because of the consecutiveWritersLimit
int consecutiveWriters = 0; // Number of writers writing in succession
int consecutiveWritersLimit = 10; // Limit on number of writers writing in succession
                                  // to prevent the writers from starving the readers

public void readData(){
 startReading();          // Ask for permission to read
 ...                      // Read the data
 stopReading();           // Signal that you are done
}

public void writeData(){
 startWriting();          // Ask for permission to write
 ...                      // Write the data
 stopWriting();           // Signal that you are done
}
```

Multiple Readers/Consecutive Writers Monitor

```
public synchronized void startReading(){  // Enter synchronized method  before reading
  readers = readers + 1;                  // I want to read
  while (writers > 0) {                   // Check if a writer is writing or waiting to write
    try {                                 // up to the consecutiveWritersLimit
      wait();                             // If so, wait
    }
    catch (InterruptedException e) {}     // Wakeup
  }                                       // Go back and try again
  reading = reading + 1;                  // I'm now reading
  consecutiveWriters = 0;                 // No more writers writing consecutively
                                          // Prevents the readers from starving the writers
}

public synchronized void stopReading(){  // Enter synchronized method after reading
  readers = readers – 1;                  // I'm no longer reading or waiting to read
  reading = reading –  1;                 // I'm done reading
  if (reading == 0) then                  // Notify waiting writer if no readers are reading
     notifyAll();
}
```

Multiple Readers/Consecutive Writers Monitor

```
public synchronized void startWriting(){  // Enter synchronized method before writing
  while ((readers > 0) and (consecutiveWriters >= consecutiveWritersLimit))
  try {
    wait();                                  // Wait here to give the readers a chance
                                             // While waiting here, I am not counted in writers
  }
  catch (InterruptedException e) {}
  }
  writers = writers + 1;                     // I want to write; excludes future readers
  while ((writing > 0) or                    // Check if writer is writing or
         (reading > 0)) {                    //  readers are reading
   try {
     wait();                                 // If so, wait
   }
   catch (InterruptedException e) {}         // Wakeup
 }                                           // Go back and try again
 writing = writing + 1;                      // I'm now writing; excludes other writers
 consecutiveWriters = consecutiveWriters + 1; // I'm the next consecutive writer writing
}
```

Multiple Readers/Consecutive Writers Monitor

```
public synchronized void stopWriting(){  // Enter synchronized method after writing
  writers = writers – 1;                 // I'm no longer writing or waiting to write
  writing = writing – 1;                 // I'm no longer writing; no writers are writing
  notifyAll();                           // Notify other readers and writers
}
```