Condition Variables Exercises

Condition Variable Scenario

- In this scenario, the processing thread creates a unique_lock instance to lock the mutex, but the fetching thread creates a lock_guard
- Why do the two threads use different lock types?

Condition Variable Example

- Write a program to test the code given in the lecture
  - The main function starts a reader thread and a writer thread, in that order
- Check that the program compiles and runs as expected
- Now reverse the order of the threads, so that the writer thread is started first. Add a sleep (say, half a second) before starting the reader thread. What happens?
- (The code is reproduced in the next two slides)

Condition Variable Example

```
// Global variables
condition_variable cv;                          // The condition variable instance
mutex mut;                                      // The mutex used to protect the data
string sdata {"Empty"};                         // The shared data

// Waiting thread
void reader() {
   unique_lock<std::mutex> guard(mut);          // Acquire lock
   cv.wait(guard);                              // Unlock mutex and wait to be notified
   // Notification received                     // Wake up and lock mutex
   cout << "Data is " << sdata << endl;         // Use the new value
}
```

Condition Variable Example

```
// Modyifing thread
void writer() {
    cout << "Writing data..." << endl;
    std::this_thread::sleep_for(2s);                    // Pretend to be busy...
    {
        lock_guard<std::mutex> lg(mut);                 // Acquire lock
        sdata = "Populated";                            // Modify the shared data
    }                                                   // Release the lock
    cv.notify_one();                                    // Notify the condition variable
}                                                       // Release the mutex
```