# Concurrent Object Oriented Languages

## java.util.concurrent.locks

wiki.eecs.yorku.ca/course/6490A

# java.util.concurrent.locks

The package java.util.concurrent.locks contains the iterfaces

- Condition
- Lock
- ReadWriteLock

# Lock

The interface Lock is implemented by the classes

- ReentrantLock
- ReentrantReadWriteLock.ReadLock
- ReentrantReadWriteLock.WriteLock

It provides more flexibility than synchronized methods and synchronized blocks.

# Lock

The Lock interface contains the methods

- lock(): acquire this lock
- unlock(): release this lock
- newCondition(): returns a condition variable bound this lock

# Lock chaining

```
Node parent = null;
Node node = this.getRoot();
node.lock()
while (!node.isLeaf())
{
    parent = node;
    node = node.getLeft();
    node.lock();
    parent.unlock();
}
node.unlock();
```

# Locks and Exceptions

```
Lock lock = ...;
lock.lock();
try
{
   ...
}
finally
{
   lock.unlock();
}
```

# Condition

The Condition interface contains the methods

- await(): causes the current thread to wait on this condition
- signal(): wakes up one thread waiting on this condition
- signalAll(): wakes up all threads waiting on this condition

# Condition

The interface Condition is implemented by the classes

- AbstractQueuedLongSynchronizer.ConditionObject
- AbstractQueuedSynchronizer.ConditionObject

# The producer-consumer problem

**Problem**

Implement the class BoundedBuffer and its methods put and get using Locks and Conditions.

# ReadWriteLock

The interface ReadWriteLock contains the methods

- readLock(): the lock used for reading
- writeLock(): the lock used for writing

# ReadWriteLock

The interface ReadWriteLock is implemented by the class ReentrantReadWriteLock.

# The readers-writers problem

**Problem**

Implement the class Database and its methods read and write using ReadWriteLocks.