

```
// an implementation with java 1.5 & above
class BarberShop {
    final Lock lock = new ReentrantLock();
    final Condition bAvail = lock.newCondition();        // signaled when barber > 0
    final Condition chOccupied = lock.newCondition();    // signaled when chair > 0
    final Condition dOpen = lock.newCondition();         // signaled when open > 0
    final Condition cuLeft = lock.newCondition();        // signaled when open = 0

    int barber = 0,    // incremented by barber when he's ready
        chair = 0,    // incremented by customer when sitting in chair
        open = 0;     // incremented by barber, decremented by cust. when leaving

    // called by customers
    public void getHaircut() {
        lock.lock();

        try {
            while (barber == 0) bAvail.await();
            barber--; chair++; chOccupied.signal();
            while (open == 0) dOpen.await();
            open--;
            cuLeft.signal();
        }
        catch (InterruptedException e) { e.printStackTrace(); }
        finally {
            lock.unlock();
        }
    }
}
```

```
// called by the barber
public void getNextCustomer() {
    lock.lock();

    try {
        barber++; bAvail.signal();
        while (chair == 0) chOccupied.awaitUninterruptibly();
        chair--;
    }
    finally {
        lock.unlock();
    }
}
```

```
// called by the barber
public void finishedCut() {
    lock.lock();

    try {
        open++; dOpen.signal();
        while (open > 0) cuLeft.awaitUninterruptibly();
    }
    finally {
        lock.unlock();
    }
}
```

```
}
```

```

public class BarberShop {

    private int barber = 0, // ++ when barber arrives, -- when client is in chair
               chair = 0,   // ++ when client goes in chair, -- when getting cut
               open = 0;    // ++ when barber opens door, -- when client left

    //_condvar(barber_available)           // signaled when the barber > 0
    //_condvar(chair_occupied)             // signaled when chair > 0
    //_condvar(door_open)                  // signaled when open > 0
    //_condvar(customer_left)              // signaled when open = 0

    // called by customers when they want service
    public synchronized void get_haircut() {
        try {
            while (barber == 0 ) {
                wait(); // wait(barber_available);
            }
            barber= barber - 1;  chair= chair + 1;

            notifyAll(); // signal(chair_occupied)
            while (open == 0 ) {
                wait(); //wait(door_open);
            }
            open = open - 1;
            notifyAll(); // signal(customer_left)
        }
        catch(IllegalMonitorStateException imse) { imse.printStackTrace(); }
        catch(InterruptedException ie) { ie.printStackTrace(); }
    }
}

```

```

    // called by barber
public synchronized void get_next_customer() {
    try {
        barber= barber + 1;
        notifyAll();          // signal(barber_available)
        while (chair == 0 ) {
            wait();          // wait(chair_occupied);
        }
        chair= chair - 1;
    }
    catch(IllegalMonitorStateException imse) { imse.printStackTrace(); }
    catch(InterruptedException ie) { ie.printStackTrace(); }
}

```

```

// called by barber
public synchronized void finished_cut() {
    try {
        open= open + 1;
        notifyAll(); //signal(door_open)
        while (open == 0 ) {
            wait();          //wait(customer_left);
        }
    }
    catch(IllegalMonitorStateException imse) { imse.printStackTrace(); }
    catch(InterruptedException ie) { ie.printStackTrace(); }
}

```

```

}

```