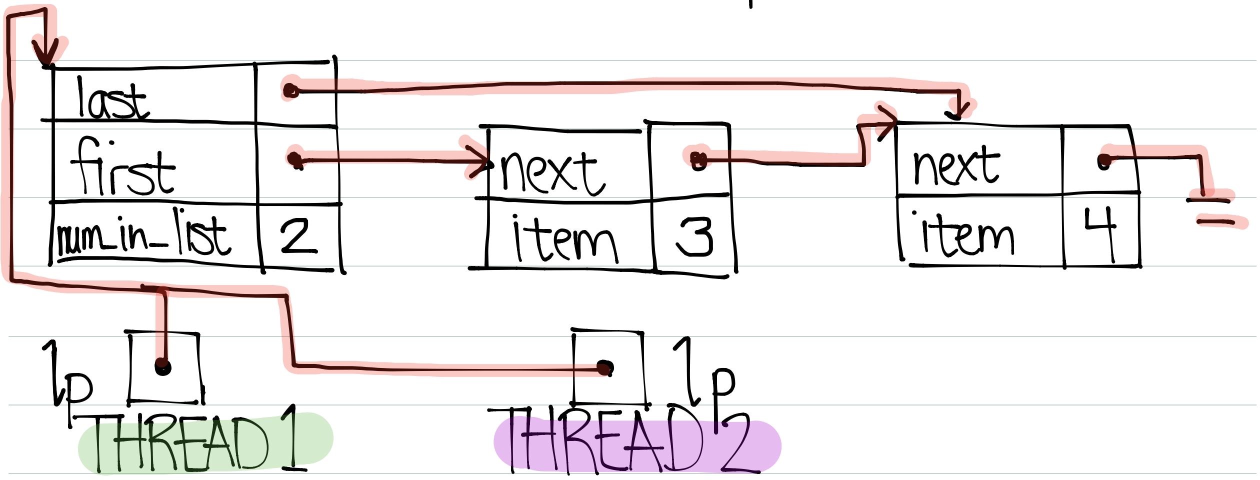
sys161.conf - can change # of processors

mutual exclusion - only one thread at a time is allowed to access an object

critical section - where there are shared objects

atomic operation - indivisible: "happens or doesn't"

int list_remove_front(list*lp)



The keyword volatile will tell the compiler that there are other threads that could access and alter a variable. Used to declare shared variables.

If volatile is not used, the compiler will optimize the code such that some code is not run.

On a uniprocessor disabling interrupts will enable mutual exclusion.

Thread 0

```
flag[0]=true
turn= 1
while(flag [1] && turn==1) { }

/* critical section */

flag[0]=false
```

Thread 1

```
flag[1]=true
turn=0
while(flag[0]&&turn==0) { }

/* critical section */

flag[1]=false;
```