3 problems. 40 points. 30 minutes *Closed book. Closed notes. No electronic device.* *Write your name above.*

1. [10 points]

Requests are issued to an io device at the rate of 100 requests/second. The requests are of two types: $1/3$ are type 1 and the remainder are type 2. The average completion time (from entry to departure) over all requests is 55 ms (milliseconds). The average completion time for type 1 requests is 33 ms.

a. What is the average number of requests (of both types) at the io device (both waiting and served)? Explain briefly.

b. What is the average number of type 2 requests at the io device. Explain briefly.

2. [10 points] Here is an attempted 2-user spinlock for a multi-cpu system. Does it ensure that at most user holds the lock at any time? If yes, explain very briefly. If no, give an evolution ending with both users holding the lock.

```
Lock:
   flag[0], flag[1]: initially false
   turn: initially 0

rel():
   flag[myid] ← false
```

```
acq():
   j ← 1 - myid
   turn ← j
   flag[myid] ← true
   while (flag[j] and turn = j) skip
```

3. [20 points] Here is a skeleton (multi-cpu) implementation of a condition variable `cv` associated with lock `lck`. Complete the implementation by filling in the boxes. Do not change what is already given.

Variables

- `runq`: run queue
- `runqL`: spinlock protecting `runq`
- `readyq`: ready queue
- `readyqL`: spinlock protecting `readq`
- `lck`: lock associated with `cv`
- `cvq`: pcb queue for `cv`
- `cvqL`: spinlock protecting `cvq`
- supply other static variables if needed

Functions

- `scheduler()`: assumes `runqL` and `readyqL` are free when called
- `updateRunqPcb()`
- `cv.signal()`:

```
cvqL.acq()
if (cvq not empty)
   readyqL.acq()
   move a pcb from cvq to readyq
   readyqL.rel()
cvqL.rel()
```

- `cv.wait()`:

```
supply code
```