CS241 #17. Producer Consumer, Semaphores, Condition Variables. Barriers & Reader Writer Problem

1. Producer Consumer & Counting Semaphores (review)

Assume buffer is an array of length 16.

```
01  void add(value) {
02   sem_wait(&sem_empty)
03   buffer[ (in++) & 15 ] = value;
04   sem_post(&sem_full);
05  }
```

```
06  remove() {
07   sem_wait(&sem_full);
08   result = buffer[ (out++) & 15 ];
09   sem_post(&sem_empty);
10   return result;
11  }
```

Q. What are 'sem_empty' and sem_full? When do they block?

Q. What should be their initial values?

Q. What if sem_full was only initialized to 7? Would the producer consumer still work? initialized to 32?

Q. What is missing from the above code? When would it matter?

Q. Could you implement a producer consumer queue using condition variables instead?

2. Fix the following multithread code to be thread safe, and use condition variables to avoid busy waiting

```
01  #define N (8)

02  pthread_cond_t cvs[N];
03  pthread_mutex_t locks[N];
04  int data[N+1];
05  int quit;

06  void init() {
07    for(int i =0; i < N;i++) {
08      data[i] = random() % N;
09
10      pthread_cond_init(cvs + i, NULL);
11      pthread_mutex_init(locks + i, NULL);
12    }
13  }

14  // Waits until data[i] > 1, then subtract 2 and increment data[i+1]
15  void runner(int i) {
16     while(!quit) {
17       while(data[i] < 2) {
18          sleep for a bit
19       }
20       data[i] -= 2;
21       data[i+1] ++;
22     }
23  }
24
25  int modify(int index, int amount) {


26    data[index] += amount;


27    return resources;
28  }
```

3.Counting Semaphore Quick Review I: sem_post {will always / may / will } never block.

3.Counting Semaphore Quick Review II

10 threads call `sem_wait`. 3 threads immediately continue, the other 7 are blocked. Then `sem_post` is called twice. How many additional threads will continue?

4. Three classic / well known synchronization problems:

Barrier

Producer Consumer

Reader-Writer Problem

5. pthread barriers

```
pthread_barrier_init( &barrier, _______);
pthread_barrier_destroy(&barrier)

pthread_barrier_wait( &barrier)
```

Return values?

```
0
PTHREAD_BARRIER_SERIAL_THREAD
```

6. Use a CV to implement a single-use barrier until all 8 threads have reached the barrier.

7. Challenge:
Make a barrier using only counting semaphores?
Make a barrier using only mutex locks?