Names:

In-class work 03/16/2015

Study the producer-consumer solution below.

1. Does it work for 1 consumer, 1 producer and Max=1?
2. Does it work for multiple consumers, multiple producers and Max > 1?
3. Does it work for multiple consumers, multiple producers and Max=1?

```
1   int buffer[MAX];
2   int fill = 0;
3   int use  = 0;
4
5   void put(int value) {
6       buffer[fill] = value;     // line f1
7       fill = (fill + 1) % MAX; // line f2
8   }
9
10  int get() {
11      int tmp = buffer[use];   // line g1
12      use = (use + 1) % MAX;   // line g2
13      return tmp;
14  }
```

Figure 31.5: The Put and Get Routines

```
1   sem_t empty;
2   sem_t full;
3
4   void *producer(void *arg) {
5       int i;
6       for (i = 0; i < loops; i++) {
7           sem_wait(&empty);           // line P1
8           put(i);                     // line P2
9           sem_post(&full);            // line P3
10      }
11  }
12
13  void *consumer(void *arg) {
14      int i, tmp = 0;
15      while (tmp != -1) {
16          sem_wait(&full);            // line C1
17          tmp = get();                // line C2
18          sem_post(&empty);           // line C3
19          printf("%d\n", tmp);
20      }
21  }
22
23  int main(int argc, char *argv[]) {
24      // ...
25      sem_init(&empty, 0, MAX); // MAX buffers are empty to begin with...
26      sem_init(&full, 0, 0);    // ... and 0 are full
27      // ...
28  }
```

Names:

Is the solution below correct:

```
1   sem_t empty;
2   sem_t full;
3   sem_t mutex;
4
5   void *producer(void *arg) {
6       int i;
7       for (i = 0; i < loops; i++) {
8           sem_wait(&mutex);           // line p0 (NEW LINE)
9           sem_wait(&empty);           // line p1
10          put(i);                     // line p2
11          sem_post(&full);            // line p3
12          sem_post(&mutex);           // line p4 (NEW LINE)
13      }
14  }
15
16  void *consumer(void *arg) {
17      int i;
18      for (i = 0; i < loops; i++) {
19          sem_wait(&mutex);           // line c0 (NEW LINE)
20          sem_wait(&full);            // line c1
21          int tmp = get();            // line c2
22          sem_post(&empty);           // line c3
23          sem_post(&mutex);           // line c4 (NEW LINE)
24          printf("%d\n", tmp);
25      }
26  }
27
28  int main(int argc, char *argv[]) {
29      // ...
30      sem_init(&empty, 0, MAX); // MAX buffers are empty to begin with...
31      sem_init(&full, 0, 0);    // ... and 0 are full
32      sem_init(&mutex, 0, 1);   // mutex=1 because it is a lock (NEW LINE)
33      // ...
34  }
```