```c
#include <stdlib.h>
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
#define N 10
#define LOOPS 20

/* Lock */
pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;
/* counts empty buffer slots */
sem_t empty;
/* counts full buffer slots */
sem_t full;

/* shared data */
char buffer[N];
int result = 0, idx = 0;

/* Threads to run */
void producer(void);
void consumer(void);

int main(int argc, char *argv[])
{
        pthread_t t1, t2;
        sem_init(&empty,0,N); sem_init(&full, 0, 0);

        pthread_create(&t1, NULL,
                (void *)producer , (void *) NULL);
        pthread_create(&t2, NULL,
                (void *)consumer , (void *) NULL);

        pthread_join(t1,NULL);
        pthread_join(t2,NULL);

        printf("Result = %d\n", result);
        return 0;
}
```

```
void producer(void){
        int item, i=LOOPS;

        while (i--) {
                /* Produce item */
                item = 1 + (int)(20.0 * rand()/(RAND_MAX+1.0));
                /* decrement empty count */
                sem_wait(&empty);
                /* enter critical region */
                pthread_mutex_lock(&mutex);

                printf("PRODUCER: idx=%d item=%d\n", idx, item);
                buffer[idx++] = item;

                /* leave critical region */
                pthread_mutex_unlock(&mutex);
                /* increment count of full slots */
                sem_post(&full);
        }
}

void consumer(void){
        int item, i=LOOPS;

        while (i--) {
                /* decrement full count */
                sem_wait(&full);
                /* enter critical region */
                pthread_mutex_lock(&mutex);

                item = buffer[--idx];
                printf("CONSUMER: idx=%d item=%d\n", idx, item);

                /* leave critical region */
                pthread_mutex_unlock(&mutex);
                /* increment count of empty slots */
                sem_post(&empty);

                /* Consume item */
                result += item;
        }
}
```