```c
#include <pthread.h>      /* standard lines */
#include <semaphore.h>
#define SHARED 1
#include <stdio.h>

void *Producer(void *);  /* the two threads */
void *Consumer(void *);

sem_t empty, full;       /* global semaphores */
int data;                /* shared buffer     */
int numIters;

/* main() -- read command line and create threads */
int main(int argc, char *argv[]) {
  pthread_t pid, cid;     /* thread and attributes */
  pthread_attr_t attr;    /* descriptors           */
  pthread_attr_init(&attr);
  pthread_attr_setscope(&attr, PTHREAD_SCOPE_SYSTEM);

  sem_init(&empty, SHARED, 1);  /* sem empty = 1 */
  sem_init(&full, SHARED, 0);   /* sem full = 0  */

  numIters = atoi(argv[1]);
  pthread_create(&pid, &attr, Producer, NULL);
  pthread_create(&cid, &attr, Consumer, NULL);
  pthread_join(pid, NULL);
  pthread_join(cid, NULL);
}

/* deposit 1, ..., numIters into the data buffer */
void *Producer(void *arg) {
  int produced;
  for (produced = 1; produced <= numIters; produced++) {
    sem_wait(&empty);
    data = produced;
    sem_post(&full);
  }
}

/* fetch numIters items from the buffer and sum them */
void *Consumer(void *arg) {
  int total = 0, consumed;
  for (consumed = 1; consumed <= numIters; consumed++) {
    sem_wait(&full);
    total = total + data;
    sem_post(&empty);
  }
  printf("the total is %d\n", total);
}
```

Figure 4.15 Simple producer/consumer using Pthreads.