```c
#include <pthread.h>
#include <semaphore.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

#define BUFFER_SIZE 10
#define PRODUCER_COUNT 1
#define CONSUMER_COUNT 1
#define PRODUCE_COUNT 20

int buffer[BUFFER_SIZE];
int in = 0;
int out = 0;

sem_t empty;
sem_t full;
pthread_mutex_t mutex;

void *producer(void *arg) {
  for (int i = 0; i < PRODUCE_COUNT; i++) {
    sem_wait(&empty);
    pthread_mutex_lock(&mutex);

    buffer[in] = i;
    printf("Produced: %d ", buffer[in]);
    in = (in + 1) % BUFFER_SIZE;

    pthread_mutex_unlock(&mutex);
    sem_post(&full);

    usleep(rand() % 100000); // Simulate variable production time
  }
  return NULL;
}

void *consumer(void *arg) {
  for (int i = 0; i < PRODUCE_COUNT; i++) {
    sem_wait(&full);
    pthread_mutex_lock(&mutex);

    int item = buffer[out];
    printf("Consumed: %d\n", item);
    out = (out + 1) % BUFFER_SIZE;

    pthread_mutex_unlock(&mutex);
    sem_post(&empty);

    usleep(rand() % 1000000); // Simulate variable consumption time
  }
  return NULL;
}

int main() {
  pthread_t producers[PRODUCER_COUNT];
  pthread_t consumers[CONSUMER_COUNT];

  sem_init(&empty, 0, BUFFER_SIZE);
  sem_init(&full, 0, 0);
  pthread_mutex_init(&mutex, NULL);

  for (int i = 0; i < PRODUCER_COUNT; i++) {
    pthread_create(&producers[i], NULL, producer, NULL);
  }

  for (int i = 0; i < CONSUMER_COUNT; i++) {
    pthread_create(&consumers[i], NULL, consumer, NULL);
  }

  for (int i = 0; i < PRODUCER_COUNT; i++) {
    pthread_join(producers[i], NULL);
  }

  for (int i = 0; i < CONSUMER_COUNT; i++) {
    pthread_join(consumers[i], NULL);
  }
```

```
  sem_destroy(&empty);
  sem_destroy(&full);
  pthread_mutex_destroy(&mutex);

  return 0;
}
```