#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

#include <semaphore.h>

#include <unistd.h>

#include <time.h>

#define BUFFER\_SIZE 5

#define MAX\_ITEMS 10

int buffer[BUFFER\_SIZE];

sem\_t mutex, empty\_slots, filled\_slots;

void \*producer(void \*arg) {

int item;

for (int i = 0; i < MAX\_ITEMS; i++) {

item = rand() % 100 + 1;

sem\_wait(&empty\_slots);

sem\_wait(&mutex);

for (int j = 0; j < BUFFER\_SIZE; j++) {

if (buffer[j] == 0) {

buffer[j] = item;

break;

}

}

printf("Produced %d, Buffer:", item);

for (int j = 0; j < BUFFER\_SIZE; j++) {

if (buffer[j] != 0) {

printf(" %d", buffer[j]);

}

}

printf("\n");

sem\_post(&mutex);

sem\_post(&filled\_slots);

usleep(rand() % 500000 + 100000); // Sleep for a random time

}

pthread\_exit(NULL);

}

void \*consumer(void \*arg) {

int item;

for (int i = 0; i < MAX\_ITEMS; i++) {

sem\_wait(&filled\_slots);

sem\_wait(&mutex);

for (int j = 0; j < BUFFER\_SIZE; j++) {

if (buffer[j] != 0) {

item = buffer[j];

buffer[j] = 0;

break;

}

}

printf("Consumed %d, Buffer:", item);

for (int j = 0; j < BUFFER\_SIZE; j++) {

if (buffer[j] != 0) {

printf(" %d", buffer[j]);

}

}

printf("\n");

sem\_post(&mutex);

sem\_post(&empty\_slots);

usleep(rand() % 500000 + 100000); // Sleep for a random time

}

pthread\_exit(NULL);

}

int main() {

srand(time(NULL));

sem\_init(&mutex, 0, 1);

sem\_init(&empty\_slots, 0, BUFFER\_SIZE);

sem\_init(&filled\_slots, 0, 0);

pthread\_t producer\_thread, consumer\_thread;

pthread\_create(&producer\_thread, NULL, producer, NULL);

pthread\_create(&consumer\_thread, NULL, consumer, NULL);

pthread\_join(producer\_thread, NULL);

pthread\_join(consumer\_thread, NULL);

sem\_destroy(&mutex);

sem\_destroy(&empty\_slots);

sem\_destroy(&filled\_slots);

printf("All threads have finished.\n");

return 0;

}

/\*Produced 98, Buffer: 98

Consumed 98, Buffer:

Produced 45, Buffer: 45

Produced 15, Buffer: 45 15

Consumed 45, Buffer: 15

Produced 76, Buffer: 76 15

Consumed 76, Buffer: 15

Produced 70, Buffer: 70 15

Produced 44, Buffer: 70 15 44

Consumed 70, Buffer: 15 44

Produced 45, Buffer: 45 15 44

Produced 5, Buffer: 45 15 44 5

Consumed 45, Buffer: 15 44 5

Produced 64, Buffer: 64 15 44 5

Produced 30, Buffer: 64 15 44 5 30

Consumed 64, Buffer: 15 44 5 30

Consumed 15, Buffer: 44 5 30

Consumed 44, Buffer: 5 30

Consumed 5, Buffer: 30

Consumed 30, Buffer:

\*/