#include <stdio.h>

#include <pthread.h>

#include <unistd.h>

#define MAX\_ORDERS 5

int order = 0;

int orders\_taken = 0;

int orders\_prepared = 0;

pthread\_mutex\_t mutex = PTHREAD\_MUTEX\_INITIALIZER;

pthread\_cond\_t cond\_order\_taken = PTHREAD\_COND\_INITIALIZER;

pthread\_cond\_t cond\_order\_prepared = PTHREAD\_COND\_INITIALIZER;

void\* take\_orders(void\* arg) {

while (1) {

pthread\_mutex\_lock(&mutex);

while (order != 0) pthread\_cond\_wait(&cond\_order\_prepared, &mutex);

if (orders\_taken >= MAX\_ORDERS) {

pthread\_cond\_signal(&cond\_order\_taken);

pthread\_mutex\_unlock(&mutex);

break;

}

orders\_taken++;

order = orders\_taken;

printf("Order taken: %d\n", order);

pthread\_cond\_signal(&cond\_order\_taken);

pthread\_mutex\_unlock(&mutex);

sleep(1);

}

return NULL;

}

void\* prepare\_food(void\* arg) {

while (1) {

pthread\_mutex\_lock(&mutex);

while (order == 0) {

if (orders\_prepared >= MAX\_ORDERS) {

pthread\_mutex\_unlock(&mutex);

return NULL;

}

pthread\_cond\_wait(&cond\_order\_taken, &mutex);

}

printf("Preparing order: %d\n", order);

sleep(2);

orders\_prepared++;

order = 0;

pthread\_cond\_signal(&cond\_order\_prepared);

pthread\_mutex\_unlock(&mutex);

if (orders\_prepared >= MAX\_ORDERS) break;

}

return NULL;

}

int main() {

pthread\_t t1, t2;

pthread\_create(&t1, NULL, take\_orders, NULL);

pthread\_create(&t2, NULL, prepare\_food, NULL);

pthread\_join(t1, NULL);

pthread\_join(t2, NULL);

printf("All orders processed!\n");

return 0;

}