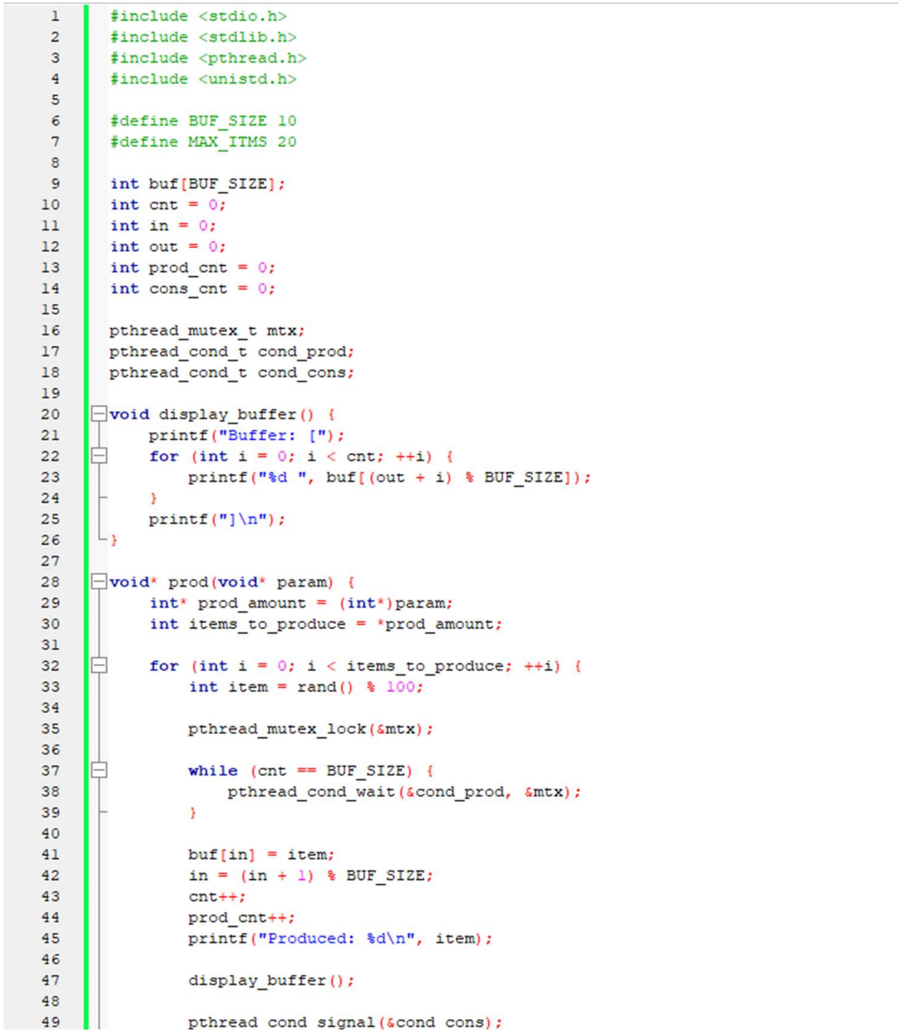
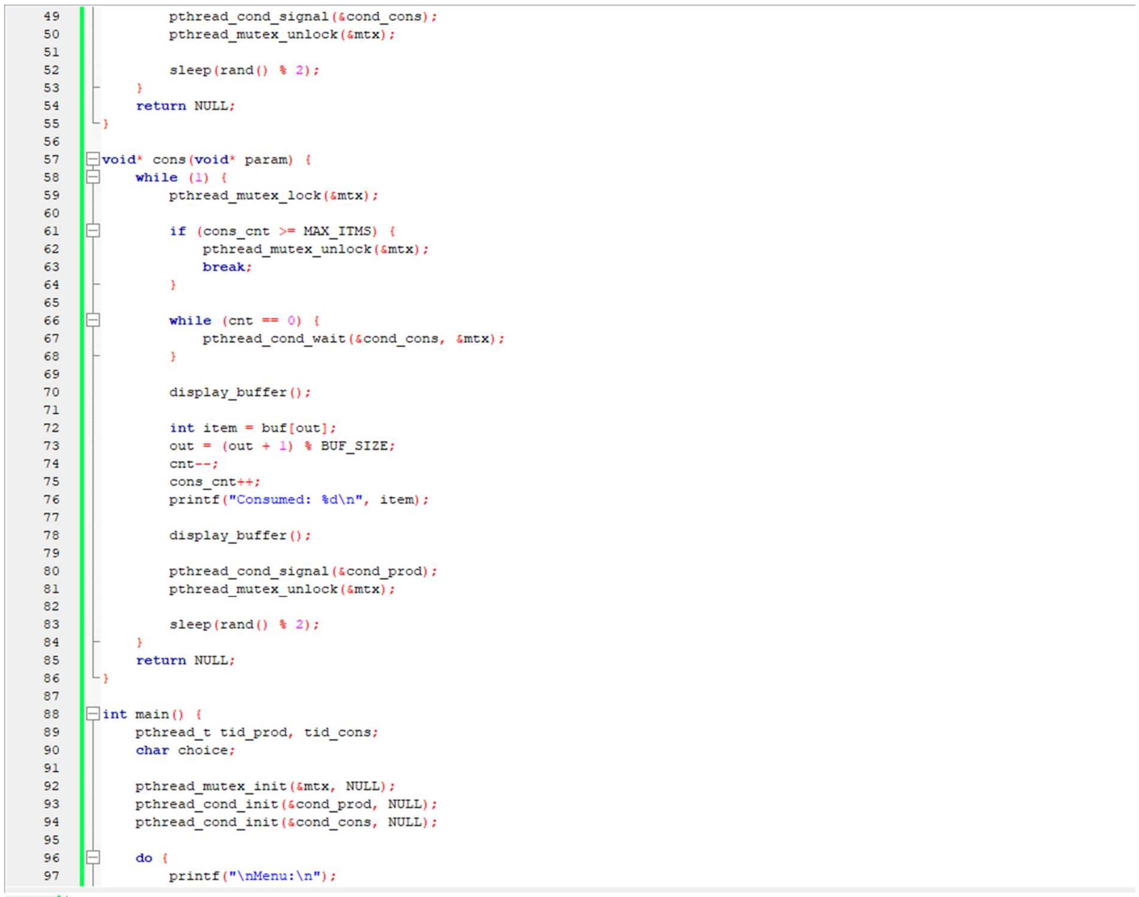
Write a C program to simulate producer-consumer problem using semaphores.

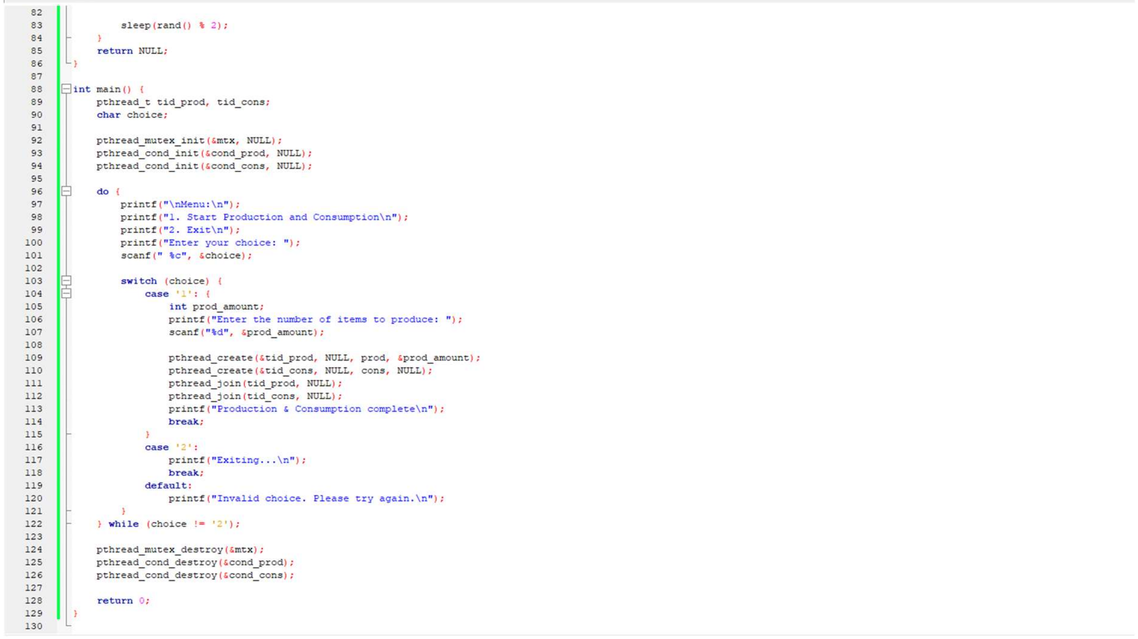
INPUT:



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

#define BUF_SIZE 10
#define MAX_ITMS 20

int buf[BUF_SIZE];
int cnt = 0;
int in = 0;
int out = 0;
int prod_cnt = 0;
int cons_cnt = 0;

pthread_mutex_t mtx;
pthread_cond_t cond_prod;
pthread_cond_t cond_cons;

void display_buffer() {
    printf("Buffer: [");
    for (int i = 0; i < cnt; ++i) {
        printf("%d ", buf[(out + i) % BUF_SIZE]);
    }
    printf("]\n");
}

void* prod(void* param) {
    int* prod_amount = (int*)param;
    int items_to_produce = *prod_amount;

    for (int i = 0; i < items_to_produce; ++i) {
        int item = rand() % 100;

        pthread_mutex_lock(&mtx);

        while (cnt == BUF_SIZE) {
            pthread_cond_wait(&cond_prod, &mtx);
        }

        buf[in] = item;
        in = (in + 1) % BUF_SIZE;
        cnt++;
        prod_cnt++;
        printf("Produced: %d\n", item);

        display_buffer();

        pthread cond signal(&cond cons);
```



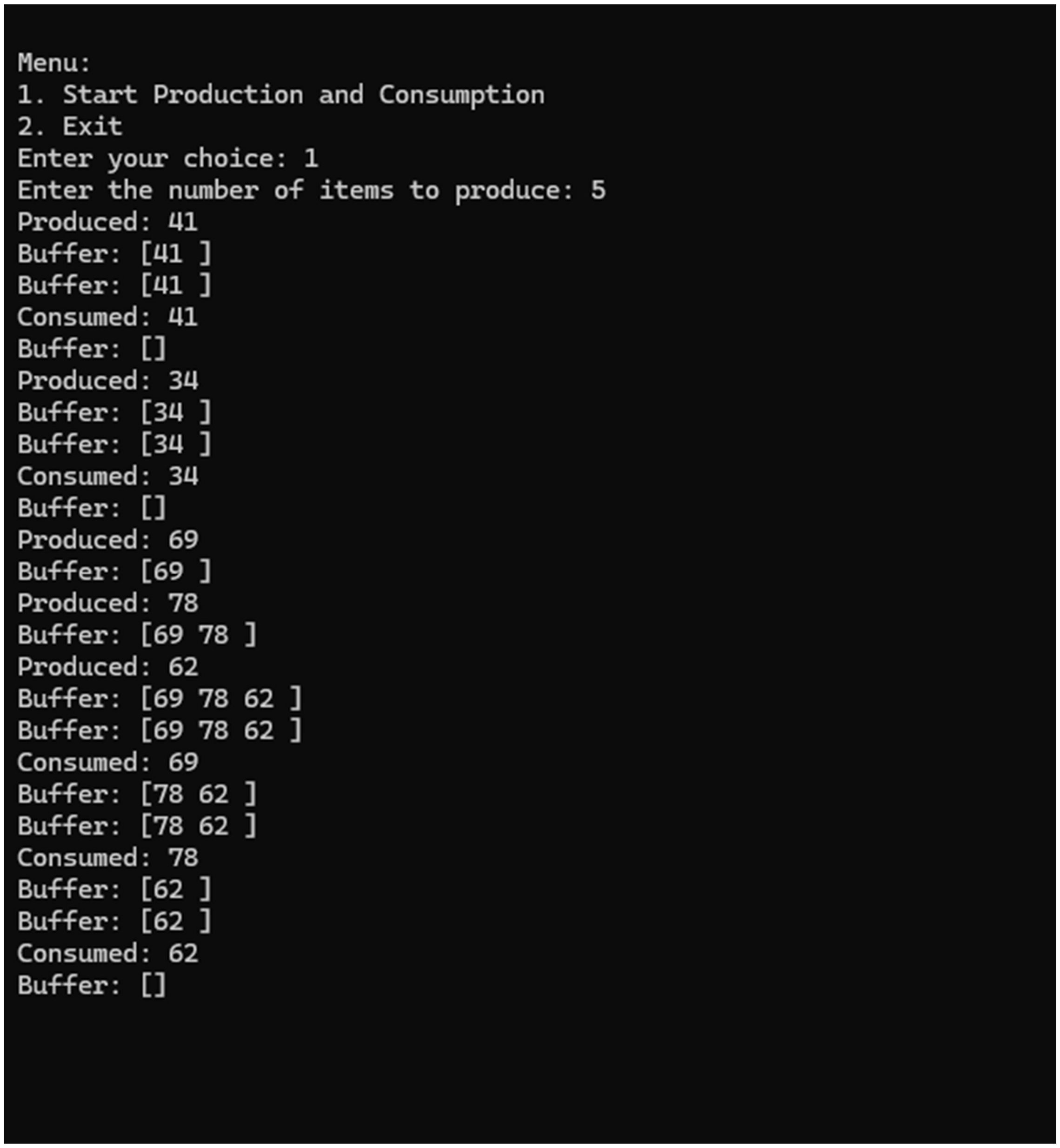
```c
        pthread_cond_signal(&cond_cons);
        pthread_mutex_unlock(&mtx);

        sleep(rand() % 2);
    }
    return NULL;
}

void* cons(void* param) {
    while (1) {
        pthread_mutex_lock(&mtx);

        if (cons_cnt >= MAX_ITMS) {
            pthread_mutex_unlock(&mtx);
            break;
        }

        while (cnt == 0) {
            pthread_cond_wait(&cond_cons, &mtx);
        }

        display_buffer();

        int item = buf[out];
        out = (out + 1) % BUF_SIZE;
        cnt--;
        cons_cnt++;
        printf("Consumed: %d\n", item);

        display_buffer();

        pthread_cond_signal(&cond_prod);
        pthread_mutex_unlock(&mtx);

        sleep(rand() % 2);
    }
    return NULL;
}

int main() {
    pthread_t tid_prod, tid_cons;
    char choice;

    pthread_mutex_init(&mtx, NULL);
    pthread_cond_init(&cond_prod, NULL);
    pthread_cond_init(&cond_cons, NULL);

    do {
        printf("\nMenu:\n");
```



```c

        sleep(rand() % 2);
    }
    return NULL;
}

int main() {
    pthread_t tid_prod, tid_cons;
    char choice;

    pthread_mutex_init(&mtx, NULL);
    pthread_cond_init(&cond_prod, NULL);
    pthread_cond_init(&cond_cons, NULL);

    do {
        printf("\nMenu:\n");
```

```c
        printf("1. Start Production and Consumption\n");
        printf("2. Exit\n");
        printf("Enter your choice: ");
        scanf(" %c", &choice);

        switch (choice) {
            case '1': {
                int prod_amount;
                printf("Enter the number of items to produce: ");
                scanf("%d", &prod_amount);

                pthread_create(&tid_prod, NULL, prod, &prod_amount);
                pthread_create(&tid_cons, NULL, cons, NULL);
                pthread_join(tid_prod, NULL);
                pthread_join(tid_cons, NULL);
                printf("Production & Consumption complete\n");
                break;
            }
            case '2':
                printf("Exiting...\n");
                break;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    } while (choice != '2');

    pthread_mutex_destroy(&mtx);
    pthread_cond_destroy(&cond_prod);
    pthread_cond_destroy(&cond_cons);

    return 0;
}
```

Output:



```
Menu:
1. Start Production and Consumption
2. Exit
Enter your choice: 1
Enter the number of items to produce: 5
Produced: 41
Buffer: [41 ]
Buffer: [41 ]
Consumed: 41
Buffer: []
Produced: 34
Buffer: [34 ]
Buffer: [34 ]
Consumed: 34
Buffer: []
Produced: 69
Buffer: [69 ]
Produced: 78
Buffer: [69 78 ]
Produced: 62
Buffer: [69 78 62 ]
Buffer: [69 78 62 ]
Consumed: 69
Buffer: [78 62 ]
Buffer: [78 62 ]
Consumed: 78
Buffer: [62 ]
Buffer: [62 ]
Consumed: 62
Buffer: []
```