main.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <semaphore.h>
#include <string.h>

#define BUFFER_SIZE 20

char *buffer;
sem_t mutex, empty, full;
int in = 0, out = 0;

void producer() {
    char item = 'a';

    while (1) {
        if (((in + 1) % BUFFER_SIZE) == out) {
            printf("Producer is waiting for the consumer...\n");
        }

        sem_wait(&empty);
        sem_wait(&mutex);

        // Load the buffer
        char current_item = item;
        buffer[in] = current_item;
        in = (in + 1) % BUFFER_SIZE;
        item = (item == 'z') ? 'a' : item + 1;

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

        printf("Produced: %c\n", current_item);
        fflush(stdout);

        sleep(1); // Produce 1 item per second
    }
}
```

main.c

```c

void consumer() {
    char item;

    while (1) {
        sem_wait(&full);
        sem_wait(&mutex);

        // Read and display the buffer
        item = buffer[out];
        out = (out + 1) % BUFFER_SIZE;

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

        printf("Consumed: %c\n", item);
        fflush(stdout);

        sleep(3); // Consume 1 item every 3 seconds
    }
}

int main() {
    int shmid;

    // Create shared memory for the buffer
    shmid = shmget(IPC_PRIVATE, BUFFER_SIZE * sizeof(char), 0666 | IPC_CREAT);
    if (shmid == -1) {
        perror("shmget");
        exit(1);
    }

    // Attach the buffer to the process
    buffer = (char *)shmat(shmid, (void *)0, 0);

    // Initialize the buffer
    for (int i = 0; i < BUFFER_SIZE; i++) {
        buffer[i] = ' ';
    }

    // Initialize semaphores
    sem_init(&mutex, 1, 1); // Mutex
```

main.c

```c
int main() {
    int shmid;

    // Create shared memory for the buffer
    shmid = shmget(IPC_PRIVATE, BUFFER_SIZE * sizeof(char), 0666 | IPC_CREAT);
    if (shmid == -1) {
        perror("shmget");
        exit(1);
    }

    // Attach the buffer to the process
    buffer = (char *)shmat(shmid, (void *)0, 0);

    // Initialize the buffer
    for (int i = 0; i < BUFFER_SIZE; i++) {
        buffer[i] = ' ';
    }

    // Initialize semaphores
    sem_init(&mutex, 1, 1); // Mutex
    sem_init(&empty, 1, BUFFER_SIZE); // Empty slots
    sem_init(&full, 1, 0); // Full slots

    if (fork() == 0) {
        producer();
    } else if (fork() == 0) {
        consumer();
    }

    // Sleep to let the child processes run
    sleep(30);

    // Clean up
    sem_destroy(&mutex);
    sem_destroy(&empty);
    sem_destroy(&full);
    shmdt(buffer);
    shmctl(shmid, IPC_RMID, NULL);

    return 0;
}
```

main.c

```
64
65 int main() {
66     int shmid;
67
```

input

```
Produced: a
Produced: b
Produced: c
Produced: d
Produced: e
Produced: f
Produced: g
Produced: h
Produced: i
Produced: j
Produced: k
Produced: l
Produced: m
Produced: n
Produced: o
Produced: p
Produced: q
Produced: r
Produced: s
Producer is waiting for the consumer...
Produced: t


...Program finished with exit code 0
Press ENTER to exit console.
```