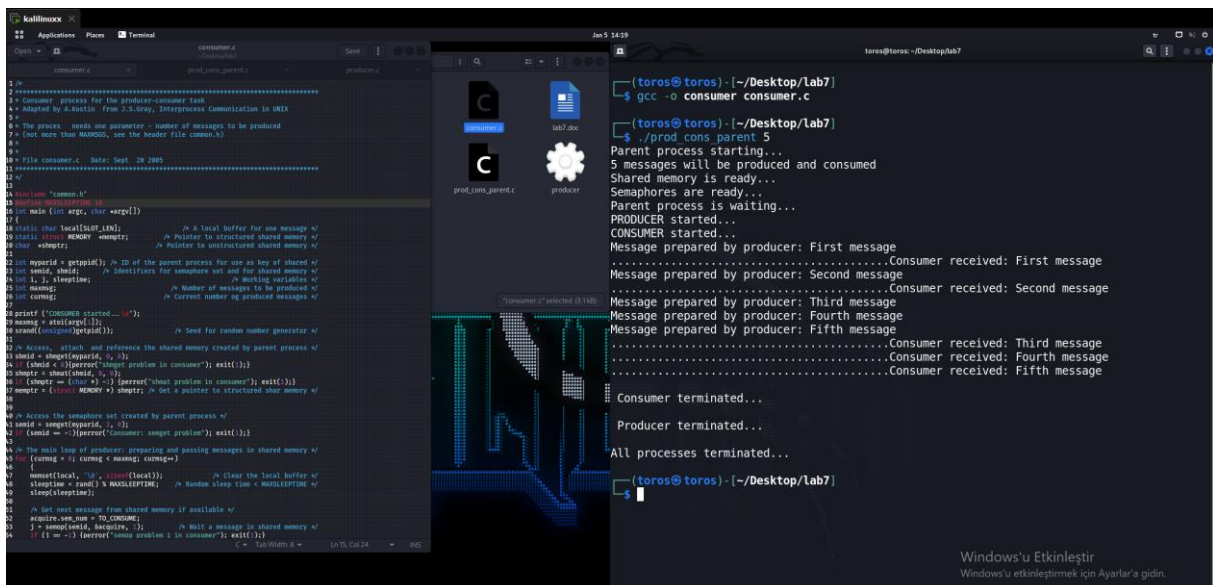
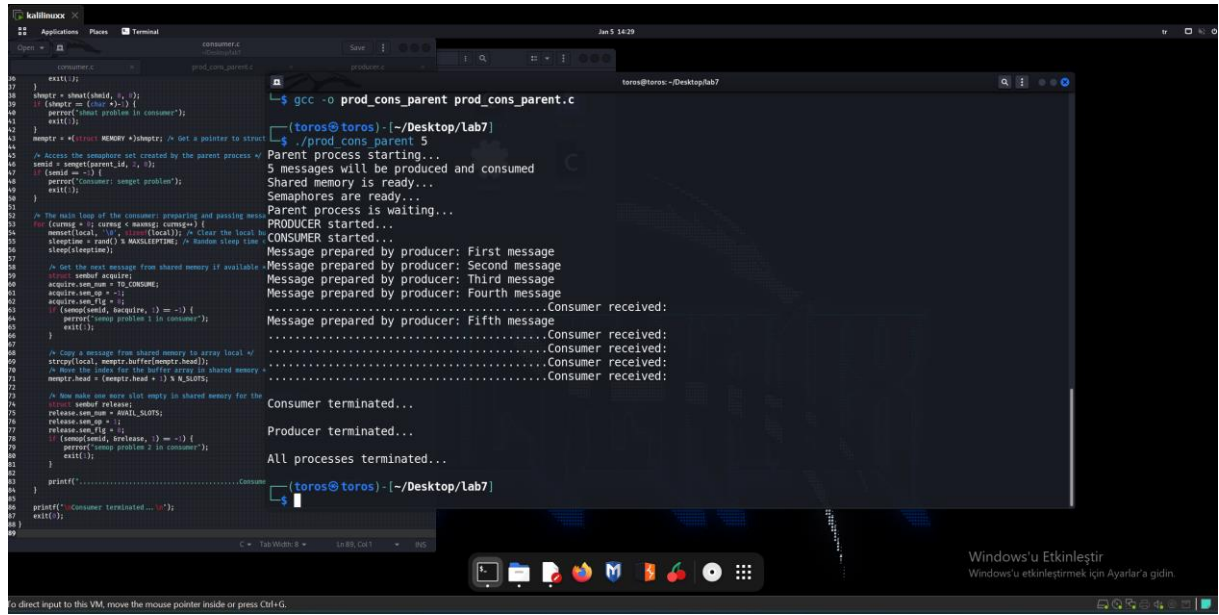


Consumer daha hızlı bir çalışıyor





```
10 exit();
11
12 shm_ptr = shmatt(shmid, 0, 0);
13 if (shm_ptr == (shm_ptr_t *) 0) {
14     perror("shmatt problem in consumer");
15     exit(1);
16 }
17
18 mem_ptr = (struct MESSAGE *)shm_ptr; // Get a pointer to struct
19
20 // Access the semaphore set created by the parent process
21 semid = semget(0, 5, 0666);
22 if (semid == -1) {
23     perror("semget: semid problem");
24     exit(1);
25 }
26
27 // The main loop of the consumer: preparing and passing mess
28 for (count = 0; count < N_MESSAGES; count++) {
29     // Clear the local buffer
30     memset(local, 0, sizeof(local));
31     // Random sleep time
32     sleep(rand() % MAX_SLEEP_TIME);
33
34     // Get the next message from shared memory if available
35     // Local semaphore acquiring
36     acquire_sem_num = 0;
37     acquire_sem_num = 0;
38     acquire_sem_num = 0;
39     if (semop(semid, sem_op, 1) == -1) {
40         perror("semop problem 1 in consumer");
41         exit(1);
42     }
43
44     // Copy a message from shared memory to array local
45     strcpy(local, mem_ptr + count * sizeof(MESSAGE));
46     // Show the index for the buffer array in shared memory
47     mem_ptr = (mem_ptr + 1) % N_MESSAGES;
48
49     // Now make one more slot empty in shared memory for the
50     // local semaphore releasing
51     release_sem_num = 0;
52     release_sem_num = 0;
53     release_sem_num = 0;
54     if (semop(semid, sem_op, 1) == -1) {
55         perror("semop problem 2 in consumer");
56         exit(1);
57     }
58
59     printf(".....Consumer\n");
60     printf("Consumer terminated... %d\n", count);
61     exit(1);
62 }
63 }
```

```
toros@toros: ~/Desktop/Lab7
$ gcc -o prod_cons_parent prod_cons_parent.c
$ ./prod_cons_parent 5
Parent process starting...
5 messages will be produced and consumed
Shared memory is ready...
Semaphores are ready...
Parent process is waiting...
PRODUCER started...
CONSUMER started...
Message prepared by producer: First message
Message prepared by producer: Second message
Message prepared by producer: Third message
Message prepared by producer: Fourth message
.....Consumer received:
Message prepared by producer: Fifth message
.....Consumer received:
.....Consumer received:
.....Consumer received:
.....Consumer received:
Consumer terminated...
Producer terminated...
All processes terminated...
toros@toros: ~/Desktop/Lab7
```

İşlem başına üretilen mesaj sayısı ve mesaj içerikleri sabit olarak tanımlanmıştır. Program, paylaşılan bellek ve semaforları oluşturduktan sonra üretici ve tüketici işlemlerini başlatır. Üretici işlemi, belirtilen sayıda rastgele mesaj üreterek kuyruğa ekler. Tüketici işlemi, üretilen mesajları kuyruktan çeker ve ekrana yazdırır.

Sonuç olarak, bu program, üretici ve tüketici işlemleri arasında paylaşılan belleği ve semaforları kullanarak basit bir iletişim modelini simüle eder. Bu tür bir model, çoklu işlemler arasında veri paylaşımı ve senkronizasyonun önemli olduğu senaryolarda kullanışlı olabilir.