EXPERIMENT 7
PRODUCER-CONSUMER PROBLEM

AIM
Implement the producer-consumer problem as a program in C.

ALGORITHM FOR PRODUCER(int *p)
STEP 0: START
STEP 1: Declare variables necessary for the program & function
STEP 2: sem-wait(&empty) ensures that producer waits if buffer is full
STEP 3: sem-wait(&mutex) ensures access to the buffer
STEP 4: a[i]=3 produces an item
STEP 5: Print which producer produced which item
STEP 6: sleep(1) simulates sleep and production time
STEP 7: buffer[i]=a[i] places the produced item in the buffer
STEP 8: sem-post(&mutex) allows other threads to access the buffer
STEP 9: sem-post(&full) signals that the buffer is filled with item.
STEP 10: END

ALGORITHM FOR CONSUMER (void *p)
STEP 0: START
STEP 1: Declare variables that are necessary for this function
STEP 2: sem_wait(&full) ensures consumer waits if buffer is empty
STEP 3: sem_wait(&mutex) ensures access to the buffer
STEP 4: Print which consumer consumed which item
STEP 5: sleep(1) simulates production time
STEP 6: b[i] = buffer[i] store the consumed item from the buffer
STEP 7: sem_post(&mutex) allows other threads to access the buffer
STEP 8: sem_post(&empty) signals that the buffer is empty.

ALGORITHM FOR MAIN
STEP 0: START
STEP 1: sem_init(&mutex, 0, 1) initializes mutex semaphore with value 1
STEP 2: sem_init(&empty, 0, 5) initializes empty semaphore with buffer size
STEP 3: sem_init(&full, 0, 0) initializes full semaphore with value 0.
STEP 4: create producer and consumer

threads using a for loop and by calling the respective functions.
STEP 5: Keep the program running indefinitely as threads handle producer-consumer operations.
STEP 6: END

RESULT
Producer-consumer problem handled successfully and output obtained.