

OS LAB -8&9

• Name :HITU RAJ

• Roll no. :2005025

• Branch :CSE

Q1-PRODUCER CONSUMER

#include <stdio.h>

#include <stdlib.h>

int mutex = 1, full = 0, empty = 3, x = 0;

int wait(int s) { return (--s); }

int signal(int s) { return (++s); }

void producer()

{

    mutex = wait(mutex);

    full = signal(full);

    empty = wait(empty);

    x++;

    printf("\nProducer produces the item %d", x);

    mutex = signal(mutex);

}

void consumer()

{

    mutex = wait(mutex);

    full = wait(full);

    empty = signal(empty);

    printf("\nConsumer consumes item %d", x);

    x--;

    mutex = signal(mutex);

}

int main()

{

    int n;

    printf("1.Producer\n2.Consumer\n3.Exit");

    while (1)

    {

        printf("\nEnter your choice : ");

        scanf("%d", &n);

        switch (n)

        {

        case 1:

            if ((mutex == 1) && (empty != 0))

                producer();

            else

                printf("Buffer is full!!");

            break;

        case 2:

            if ((mutex == 1) && (full != 0))

                consumer();

            else

                printf("Buffer is empty!!");

            break;

        case 3:

            exit(0);

            break;

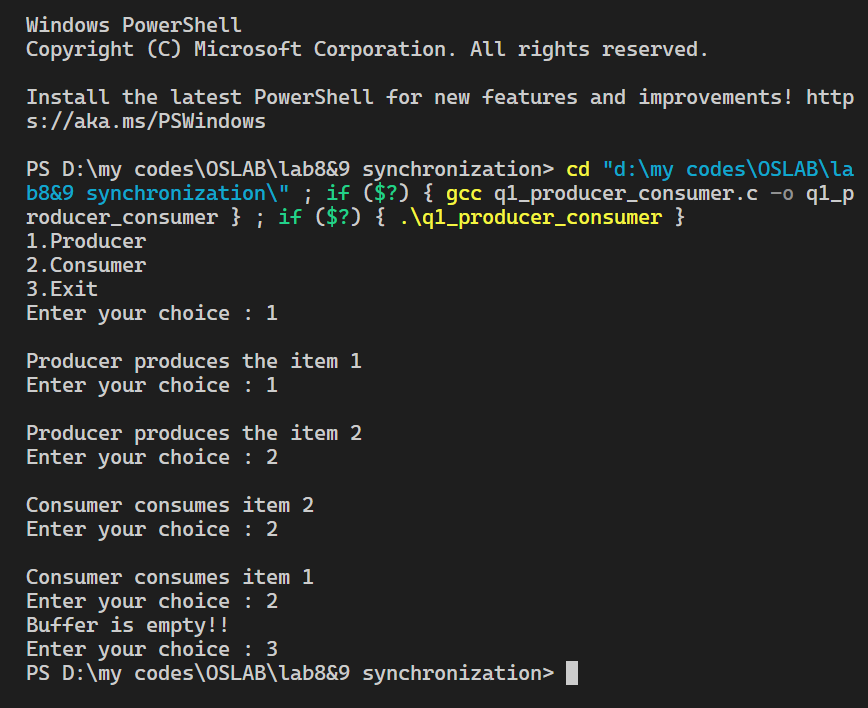
        }

    }

    return 0;

}

OUTPUT-1



Q2-READER WRITER PROBLEM

#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

#include <semaphore.h>

#include <unistd.h>

sem\_t x, y;

pthread\_t tid;

pthread\_t writerthreads[100], readerthreads[100];

int readercount = 0;

void \*reader(void \*param)

{

    sem\_wait(&x);

    readercount++;

    if (readercount == 1)

        sem\_wait(&y);

    sem\_post(&x);

    printf("%d Reader is inside\n", readercount);

    usleep(3);

    sem\_wait(&x);

    readercount--;

    if (readercount == 0)

    {

        sem\_post(&y);

    }

    sem\_post(&x);

    printf("Total readers : %d, reader is leaving\n", readercount + 1);

    return NULL;

}

void \*writer(void \*param)

{

    printf("Writer is trying to enter\n");

    sem\_wait(&y);

    printf("Writer has entered\n");

    sem\_post(&y);

    printf("Writer is leaving\n");

    return NULL;

}

int main()

{

    int n2;

    printf("Enter the number of readers/writers : ");

    scanf("%d", &n2);

    printf("\n");

    int n1[n2];

    sem\_init(&x, 0, 1);

    sem\_init(&y, 0, 1);

    for (int i = 0; i < n2; i++)

    {

        pthread\_create(&writerthreads[i], NULL, reader, NULL);

        pthread\_create(&readerthreads[i], NULL, writer, NULL);

    }

    for (int i = 0; i < n2; i++)

    {

        pthread\_join(writerthreads[i], NULL);

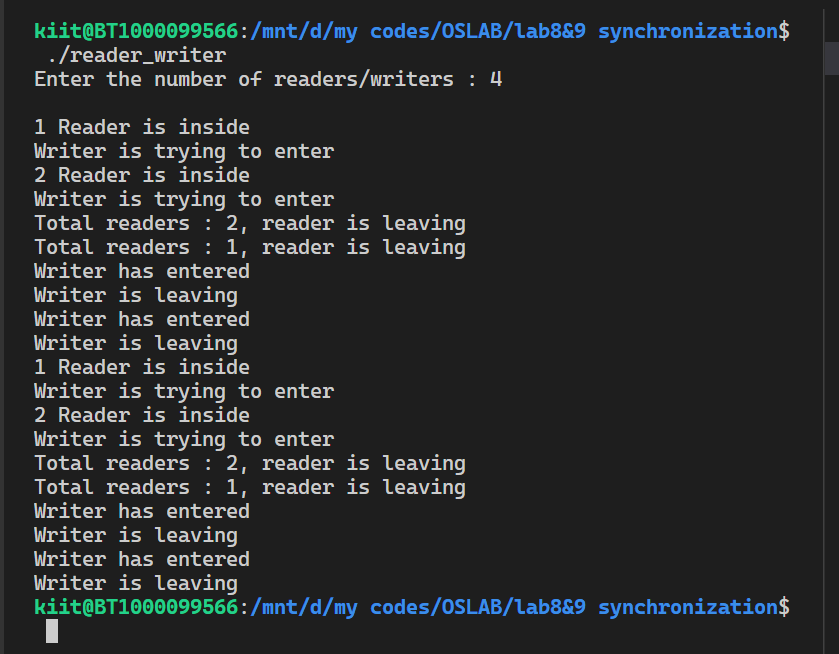
        pthread\_join(readerthreads[i], NULL);

    }

    return 0;

}

OUTPUT-2



Q3-DINING PROBLEM

#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

#include <semaphore.h>

#include <unistd.h>

sem\_t room;

sem\_t chopstick[5];

void eat(int phil) { printf("Philosopher %d is eating\n", phil); }

void \*philosopher(void \*num)

{

    int phil = \*(int \*)num;

    sem\_wait(&room);

    printf("Philosopher %d has entered thinking\n", phil);

    sem\_wait(&chopstick[phil]);

    sem\_wait(&chopstick[(phil + 1) % 5]);

    eat(phil);

    sleep(2);

    printf("Philosopher %d has finished eating\n", phil);

    sem\_post(&chopstick[(phil + 1) % 5]);

    sem\_post(&chopstick[phil]);

    sem\_post(&room);

}

int main()

{

    int i, a[5];

    pthread\_t tid[5];

    sem\_init(&room, 0, 4);

    printf("\n");

    for (i = 0; i < 5; i++)

        sem\_init(&chopstick[i], 0, 1);

    for (i = 0; i < 5; i++)

    {

        a[i] = i;

        pthread\_create(&tid[i], NULL, philosopher, (void \*)&a[i]);

    }

    for (i = 0; i < 5; i++)

        pthread\_join(tid[i], NULL);

}

OUTPUT -3

