WEEK 5 OS LAB

1BM21CS238

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

SOURCE CODE:

#include<stdio.h>

#include<stdlib.h>

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

int main()

{

int n;

void producer ();

void consumer ();

int wait(int);

int signal(int);

printf("\n1.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;

}

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);

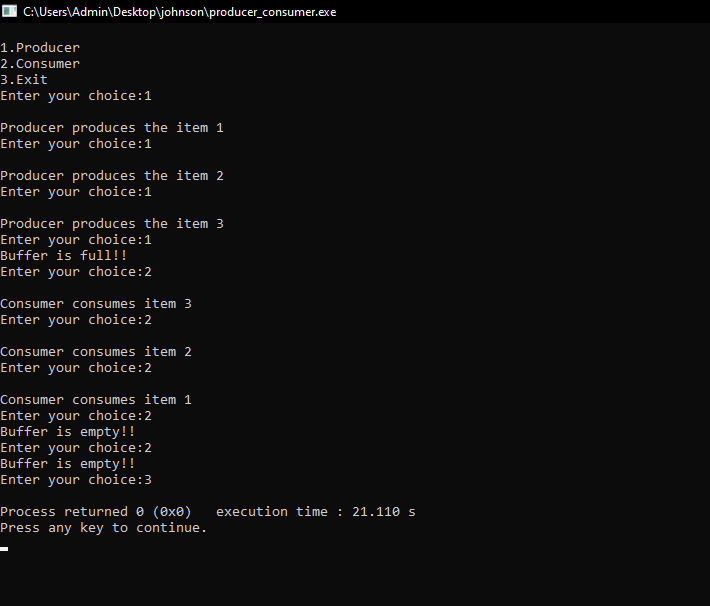
printf("\nConsumer consumes item %d",x);

x--;

mutex=signal(mutex);

}

OUTPUT:



Q)  Write a C program to simulate the concept of Dining-Philosophers problem.

SOURCE CODE:

#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

#include <semaphore.h>

#include <unistd.h>

#include<conio.h>

#define NUM\_PHILOSOPHERS 5

#define THINKING 0

#define HUNGRY 1

#define EATING 2

#define LEFT (philosopher\_num + NUM\_PHILOSOPHERS - 1) % NUM\_PHILOSOPHERS

#define RIGHT (philosopher\_num + 1) % NUM\_PHILOSOPHERS

sem\_t chopsticks[NUM\_PHILOSOPHERS];

int philosopher\_state[NUM\_PHILOSOPHERS];

void \*philosopher(void \*arg) {

int philosopher\_num = \*((int \*)arg);

while (1) {

printf("Philosopher %d is thinking.\n", philosopher\_num);

sleep(rand() % 3 + 1);

printf("Philosopher %d is hungry.\n", philosopher\_num);

philosopher\_state[philosopher\_num] = HUNGRY;

if (philosopher\_num == 0) {

sem\_post(&chopsticks[RIGHT]);

sem\_post(&chopsticks[LEFT]);

} else {

sem\_wait(&chopsticks[LEFT]);

sem\_wait(&chopsticks[RIGHT]);

}

printf("Philosopher %d is eating.\n", philosopher\_num);

sleep(rand() % 3 + 1);

printf("Philosopher %d finished eating.\n", philosopher\_num);

philosopher\_state[philosopher\_num] = THINKING;

sem\_post(&chopsticks[LEFT]);

sem\_post(&chopsticks[RIGHT]);

}

}

void main() {

pthread\_t philosophers[NUM\_PHILOSOPHERS];

int philosopher\_nums[NUM\_PHILOSOPHERS];

for (int i = 0; i < NUM\_PHILOSOPHERS; ++i) {

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

philosopher\_state[i] = THINKING;

}

for (int i = 0; i < NUM\_PHILOSOPHERS; ++i) {

philosopher\_nums[i] = i;

pthread\_create(&philosophers[i], NULL, philosopher, &philosopher\_nums[i]);

}

for (int i = 0; i < NUM\_PHILOSOPHERS; ++i) {

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

}

for (int i = 0; i < NUM\_PHILOSOPHERS; ++i) {

sem\_destroy(&chopsticks[i]);

}

getch();

}

OUTPUT:

