**OS\_LAB\_11\_Assignment**

**CE\_054**

**Aim** :- Study and implementation of Sleeping barber and Dining philosopher situations .

**Code 1 :-**

**Slepping Barber**

Code :-

// Author : Dhruv B kakadiya

#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

#include <semaphore.h>

#include <unistd.h>

#define NO\_CUSTOMERS 1000

#define NO\_BARBER 2

#define CAPACITY 5

sem\_t sem\_wait\_count, sem\_cust, sem\_barb;

int waited\_customer = 0;

void \*Customer(void \*arg)

{

    int \* custId=(int \*)arg;

    sem\_wait(&sem\_wait\_count);

    if(waited\_customer < CAPACITY)

    {

        waited\_customer++;

        printf("Customer-%d : entered wait :- \n", \*custId);

        sem\_post(&sem\_cust);

        sem\_post(&sem\_wait\_count);

        sem\_wait(&sem\_barb);

        printf("Customer-%d : got haircut :-  \n", \*custId);

    }

    else

    {

        sem\_post(&sem\_wait\_count);

    }

}

void \*Barber(void \*arg)

{

    int waitingTemp = 0;

    while(1)

    {

        sem\_wait(&sem\_cust);

        sem\_wait(&sem\_wait\_count);

        waited\_customer--;

        waitingTemp=waited\_customer;

        printf("\tBarber :-  started cutting hair \t total number of customers in waiting :- %d \n", waitingTemp);

        sem\_post(&sem\_barb);

        sem\_post(&sem\_wait\_count);

    }

}

void main()

{

    int loop, cust\_id[NO\_CUSTOMERS];

    pthread\_t barber\_arr[NO\_BARBER], customer\_arr[NO\_CUSTOMERS];

    sem\_init(&sem\_barb, 0, 0);

    sem\_init(&sem\_cust, 0, 0);

    sem\_init(&sem\_wait\_count, 0, 1);

    for(loop = 0 ; loop < NO\_BARBER ; loop++)

    {

        pthread\_create(&barber\_arr[loop], NULL,Barber, (void\*)NULL);

    }

    for(loop = 0 ; loop < NO\_CUSTOMERS ; loop++)

    {

        cust\_id[loop] = loop + 1;

        pthread\_create(&customer\_arr[loop], NULL,Customer, (void\*)&cust\_id[loop]);

    }

    for(loop = 0 ; loop < NO\_BARBER ; loop++)

    {

        pthread\_join(barber\_arr[loop], NULL);

    }

    for(loop = 0 ; loop < NO\_CUSTOMERS ; loop++)

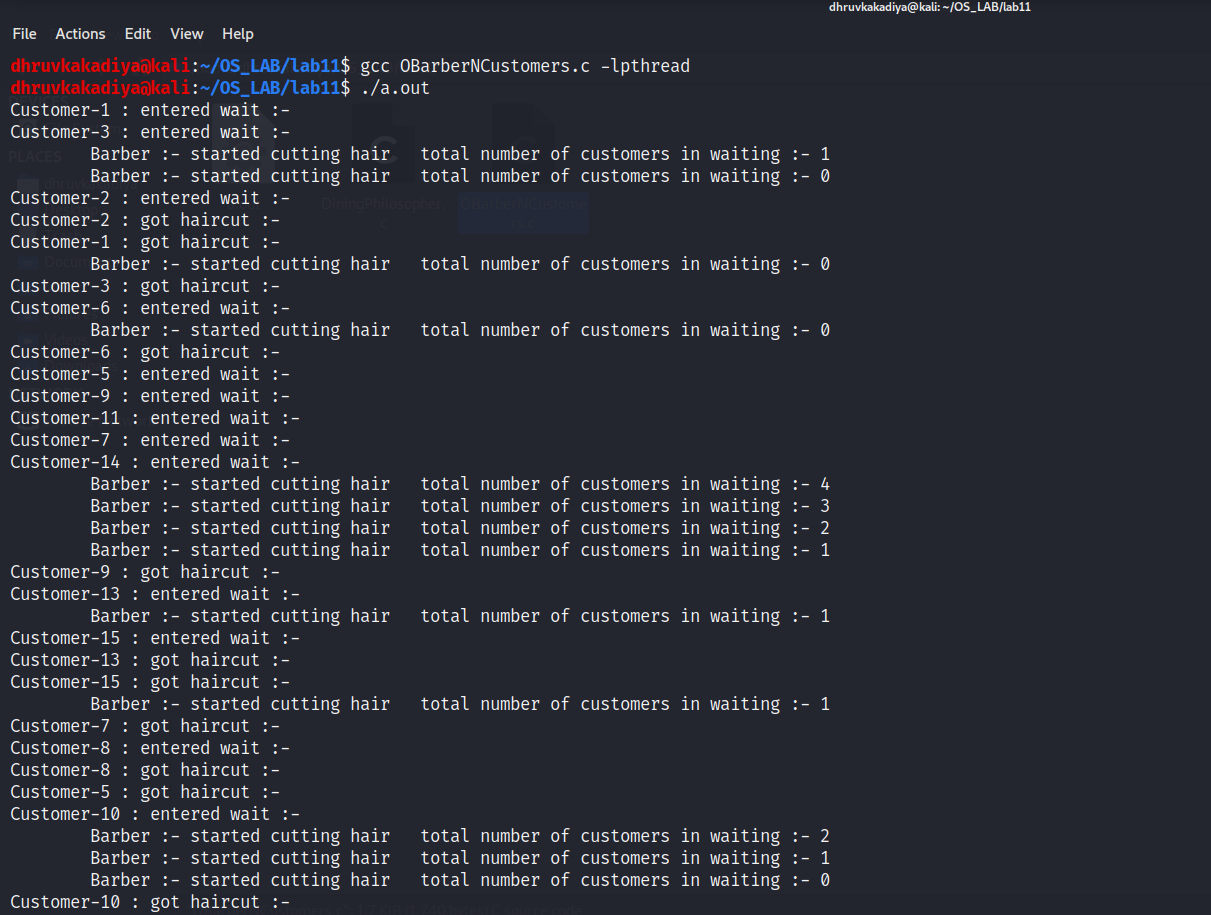
    {

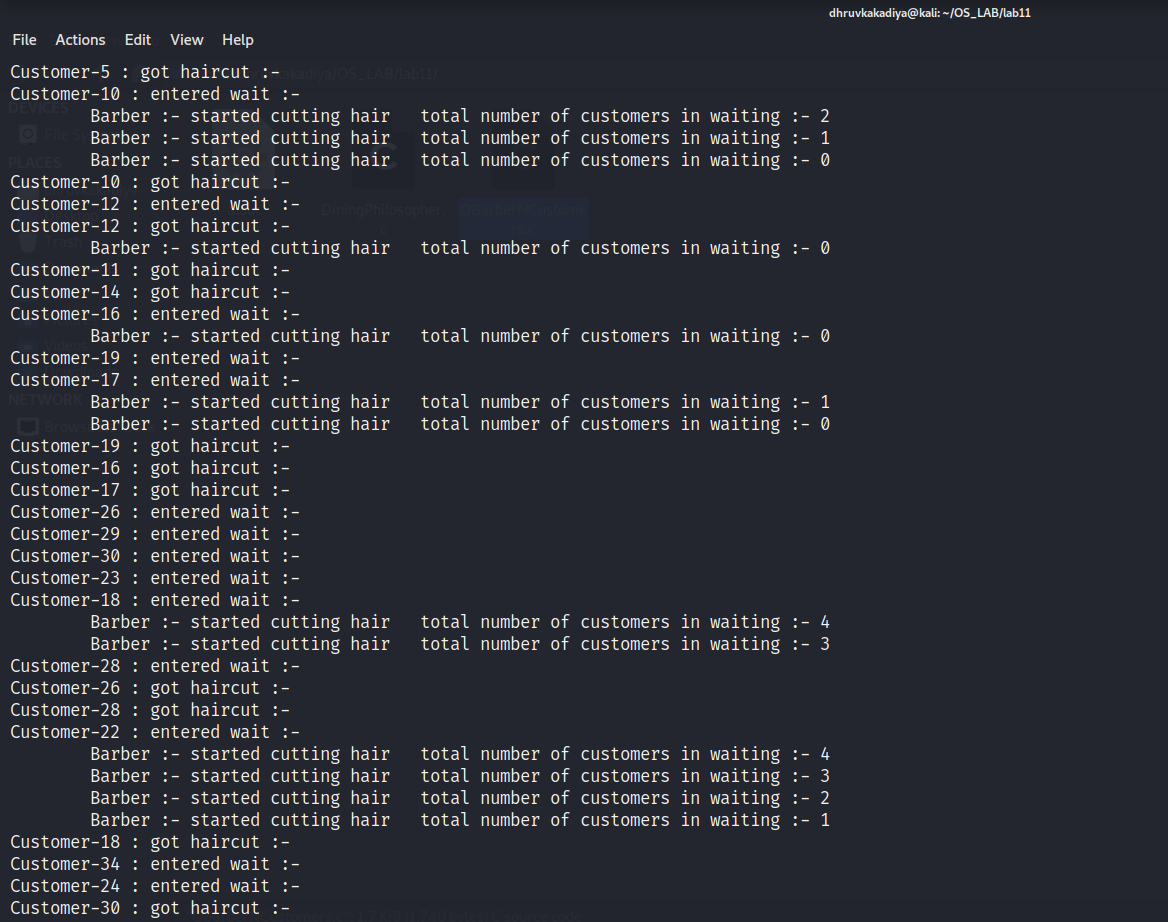
        pthread\_join(customer\_arr[loop], NULL);

    }

}

Output :-





**Code 2 :-**

**Priority -> Writer**

Code :-

// Author : Dhruv B kakadiya

#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

#include <semaphore.h>

#include <unistd.h>

#define ph\_counter 5

sem\_t sem\_fork[ph\_counter], sem\_room;

void \*Philosopher(void \*arg)

{

    int \*ph\_id = (int \*)arg;

    while(1)

    {

        sem\_wait(&sem\_room);

        printf("P :- %d thinking\n", \*ph\_id);

        sem\_wait(&sem\_fork[\*ph\_id - 1]);

        printf("\t\tP :- %d : grabed fork left\n", \*ph\_id);

        sem\_wait(&sem\_fork[(\*ph\_id)%ph\_counter]);

        printf("\tP :- %d : grabed both fork and ready to eat\n", \*ph\_id);

        sem\_post(&sem\_fork[(\*ph\_id)%ph\_counter]);

        printf("\t\tP :- %d : relesed right\n", \*ph\_id);

        sem\_post(&sem\_fork[\*ph\_id - 1]);

        printf("\t\tP :- %d : relesed fork both\n", \*ph\_id);

        sem\_post(&sem\_room);

        sleep(1);

    }

}

void main()

{

    pthread\_t phi\_thread[ph\_counter];

    int loop, ph\_id[ph\_counter];

    sem\_init(&sem\_room,0, ph\_counter-1);

    for(loop = 0 ; loop < ph\_counter ; loop++)

    {

        sem\_init(&sem\_fork[loop], 0, 1);

    }

    for(loop = 0 ; loop < ph\_counter ; loop++)

    {

        ph\_id[loop] = loop + 1;

        pthread\_create(&phi\_thread[loop], NULL, Philosopher, (void\*)&ph\_id[loop]);

    }

    for(loop = 0 ; loop < ph\_counter ; loop++)

    {

        pthread\_join(phi\_thread[loop], NULL);

    }

}

Output :-

