**OS : Practical Assignment no. 5**

***Title* :**

Write a program for implementing Synchronization-Sleeping Barber.

***Program* :**

#include <stdio.h>

#include <unistd.h>

#include <stdlib.h>

#include <time.h>

#include <pthread.h>

#include <semaphore.h>

#define MAX\_CUSTOMERS 25

void \*customer(void \*num);

void \*barber(void \*);

void randwait(int secs);

// to enter the waiting room at one time.

sem\_t waitingRoom;

// barberChair ensures mutually exclusive access to the barber chair.

sem\_t barberChair;

// barberPillow is used to allow the barber to sleep until a customer arrives.

sem\_t barberPillow;

// seatBelt is used to make the customer to wait until the barber is done cutting his/her hair.

sem\_t seatBelt;

// Flag to stop the barber thread when all customers have been serviced.

int allDone = 0;

int main(int argc, char \*argv[]) {

pthread\_t btid;

pthread\_t tid[MAX\_CUSTOMERS];

long RandSeed;

int i, numCustomers, numChairs;

int Number[MAX\_CUSTOMERS];

printf("Enter the number of Custmors : ");

scanf("%d",&numCustomers);

printf("Enter the number of Chairs : ");

scanf("%d",&numChairs);

// Make sure the number of threads is less than the number of customers we can support.

if (numCustomers > MAX\_CUSTOMERS) {

printf("The maximum number of Customers is %d.\n", MAX\_CUSTOMERS);

exit(-1);

}

// Initialize the numbers array.

for (i=0; i<MAX\_CUSTOMERS; i++) {

Number[i] = i;

}

// Initialize the semaphores with initial values…

sem\_init(&waitingRoom, 0, numChairs);

sem\_init(&barberChair, 0, 1);

sem\_init(&barberPillow, 0, 0);

sem\_init(&seatBelt, 0, 0);

// Create the barber.

pthread\_create(&btid, NULL, barber, NULL);

for (i=0; i<numCustomers; i++) {

pthread\_create(&tid[i], NULL, customer, (void \*)&Number[i]);

sleep(1);

}

for (i=0; i<numCustomers; i++) {

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

sleep(1);

}

allDone = 1;

sem\_post(&barberPillow);

pthread\_join(btid,NULL);

}

void \*customer(void \*number) {

int num = \*(int \*)number;

printf("Customer %d leaving for barber shop.\n", num);

randwait(2);

printf("Customer %d arrived at barber shop.\n", num);

sem\_wait(&waitingRoom);

printf("Customer %d entering waiting room.\n", num);

sem\_wait(&barberChair);

sem\_post(&waitingRoom);

printf("Customer %d waking the barber.\n", num);

sem\_post(&barberPillow);

sem\_wait(&seatBelt);

sem\_post(&barberChair);

printf("Customer %d leaving barber shop.\n", num);

}

void \*barber(void \*junk) {

while (!allDone) {

printf("The barber is sleeping...\n");

sem\_wait(&barberPillow);

if (!allDone) {

printf("The barber is cutting hair...\n");

randwait(2);

printf("The barber has finished cutting hair.\n\n");

sem\_post(&seatBelt);

}

else {

printf("The barber is going home for the day.\n");

}

}

}

void randwait(int secs) {

int len;

// Generate a random number…

len = (int) ((1 \* secs) + 1);

sleep(len);

}

***Output Screenshots***





