/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PROBLEM 4: DIJKSTRA'S SLEEPING BARBER PROBLEM

NAME: G A RUBAYAT HYDER

MATRIC NO.: A17CS5335

NAME: AHMAD SAMMAN

MATRIC NO.:

NAME: JOHN MATTHEW HAYUN BANAYAD

MATRIC NO.: A17CS5069

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <unistd.h>

#include <stdlib.h>

#include <time.h>

#include <pthread.h>

#include <semaphore.h>

// The maximum number of customer threads.

#define MAX\_CUSTOMERS 25

// Function prototypes

void \*customer(void \*num);

void \*barber(void \*);

void randwait(int secs); //Waits for a random amount of time

// Semaphores.

sem\_t waitingRoom;

sem\_t barberChair;

sem\_t Sleep; //Barber Sleeps

sem\_t cWait; //Customer Waits

int Complete = 0; //All Customers served and work comes to a halt

//Main

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

printf("Enter the number of Customers : "); //Prompt the required number of customers to be served

scanf("%d",&numCustomers) ;

printf("Enter the number of Chairs : "); //Prompt the number of chairs available

scanf("%d",&numChairs);

if (numCustomers > MAX\_CUSTOMERS) //The Maximum number of customers cannot exceed 25

{

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

printf("Maximum number of customers exceeded. \n");

exit(-1);

}

for (i=0; i<MAX\_CUSTOMERS; i++) //Initializing array

{

Number[i] = i;

}

//Initializing Semaphores

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

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

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

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

pthread\_create(&btid, NULL, barber, NULL); //Creating Barber

for (i=0; i<numCustomers; i++) //Creating Customers

{

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

sleep(1);

}

for (i=0; i<numCustomers; i++) // Join threads

{

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

sleep(1);

}

//When work is Complete, barber leaves

Complete = 1;

sem\_post(&Sleep); // Wake the barber so he will exit.

pthread\_join(btid,NULL);

}

void \*customer(void \*number)

{

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

printf("Customer %d makes appointment and leaves for barber shop.\n", num + 1);

randwait(2); //Waits for a random time before arriving

printf("Customer %d arrived at barber shop.\n", num + 1);

sem\_wait(&waitingRoom); //Waits for a short time for a chair to free up in the waiting room

printf("Customer %d entering waiting room.\n", num + 1);

sem\_wait(&barberChair); //Waits for Barber

sem\_post(&waitingRoom); //Customer moving to barber chair, freeing up space in waiting room

printf("Customer %d waking the barber.\n", num + 1);

sem\_post(&Sleep); //Waking up the barber

sem\_wait(&cWait); //Time required for barber to serve customer

sem\_post(&barberChair); //Customer is done, and barber chair is free

printf("Customer %d leaving barber shop.\n", num + 1);

}

void \*barber(void \*junk)

{

while (!Complete) //Barber waits while there are more appointments left

{

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

sem\_wait(&Sleep); //Barber sleeps until woken by customer

if (!Complete)

{

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

randwait(2); //Takes a random time to serve Customer

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

sem\_post(&cWait); //Customer Leaves

}

else

{

printf("The barber is going home for the day.\n"); //Barber leaves once no appointments are left

}

}

}

void randwait(int secs)

{

int i;

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

sleep(i); //Wait for random amount of time

}

Sample Output:

