Devin Hardy

CS 415

ASG 4

Code:

// Devin Hardy

// Asg 4

// Barber shop problem with 2 threads, 2 mutexes, and a condition variable

#include <iostream>

#include <random>

#include <thread>

#include <mutex>

// Synchron variables

int barberReady = 1; // IF barber is ready

int waitChairsCanBeAccessed = 0; // If customers can wait

bool cuttingHair = false;

bool closeShop = false;

int readyCustomers = 0; // # customers waiting

int NumChairs = 4; // max chairs to wait in

// global vars

int cutTime = 0;

int numServed = 0;

int numSad = 0; // Number of sad customers

// no more customers

//Random number generator

// pass in first the min number and second the max number

// for the possible random number to be between

int RandNum(int first, int second)

{

std::random\_device dev;

std::mt19937 rng(dev());

std::uniform\_int\_distribution<std::mt19937::result\_type> dist6(first, second); // distribution in range [1, 6]

return (dist6(rng));

}

// Barber Algorithm

void Barber(int runTime) {

while(!closeShop) {

if (barberReady) {

// if zero sleep

if (readyCustomers == 0) {

barberReady = 0;

}

// if not zero get customer and cut hair

else {

readyCustomers--;

barberReady = 0;

cutTime = RandNum(3, 12);

cuttingHair = true;

}

waitChairsCanBeAccessed = 1;

}

}

}

// Customer Algoithm

void Customer(int runTime) {

int getCustomer = 0;

int cstmrTick = 0;

int hairTick = 0;

int cutTime = 0;

bool addCustomer = false;

for (int i = 0; i < runTime; i++) {

if (addCustomer) {

// any seats remaining?

if (readyCustomers < NumChairs)

{

readyCustomers++;

// if barber is not cutting hair

if (cuttingHair == false) {

// and if barber is asleep then awaken them

if (barberReady == 0 && readyCustomers != 0) {

barberReady = 1;

}

}

}

else // No seats? I'm sorry

{

numSad++;

}

addCustomer = false;

}

if (cstmrTick == getCustomer) {

addCustomer = true;

getCustomer = RandNum(3, 15);

cstmrTick = 0;

}

else {

cstmrTick++;

}

if (cuttingHair) {

if (hairTick == cutTime) {

numServed++;

hairTick = 0;

cuttingHair = false;

barberReady = 1;

}

else {

hairTick++;

}

}

}

closeShop = true;

}

int main()

{

int runTime = 0;

runTime = RandNum(100, 1000);

std::thread Brbr(Barber, runTime);

std::thread Cstmr(Customer, runTime);

Brbr.join();

Cstmr.join();

std::cout << "For " << runTime << std::endl;

std::cout << "Number of customers with new haircuts = " << numServed << std::endl;

std::cout << "Number of dissappointed customers = " << numSad << std::endl;

return 0;

}

/\*

Reference:

https://techtipqa.wordpress.com/2015/08/21/tech-qa-9-the-sleeping-barber-problem/

\*/

Output:



