using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

namespace HandsOnLock\_Mutex\_Monitor

{

class MultiThreadBankingOperationsDemo

{

static void Main(string[] arg)

{

Customer customer1 = new Customer() { accountNumber = "ACT123456", customerName="Abinav", jointCustomerName="Thenmozhi", balance= 10000000.00 };

ABCBanking abcBanker = new ABCBanking(customer1);

//Console.WriteLine(abcBanker.widthDraw(10000));

//Console.WriteLine(abcBanker.widthDraw(20000));

//Console.WriteLine(abcBanker.showBalance());

Thread[] Threads = new Thread[3];

double amount = 10000.00;

for (int i = 0; i < 3; i++)

{

customer1.widthDrawAmount = amount + amount;

Threads[i] = new Thread(new ThreadStart(abcBanker.widthDraw));

Threads[i].Name = "Child " + i;

Console.WriteLine("Thead names :" + Threads[i].Name);

}

foreach (Thread t in Threads)

t.Start();

Console.ReadLine();

}

}

public class ABCBanking

{

public ABCBanking(Customer customer)

{

this.customer = customer;

}

public Customer customer { get; set; }

public void widthDraw( )

{

Monitor.Enter(this);

try

{

Console.WriteLine();

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\* Transaction Initiated \*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ");

if (this.customer.balance < 0)

{

Console.WriteLine("Negative Balance");

throw new Exception("Negative Balance");

}

if (this.customer.balance > this.customer.widthDrawAmount)

{

Thread.Sleep(1000);

Console.WriteLine("Executing Thread Name : " + Thread.CurrentThread.Name);

Console.WriteLine("Account Balance before Withdrawal : " + this.customer.balance);

Console.WriteLine("Amount to Withdraw : -" + this.customer.widthDrawAmount);

this.customer.balance = this.customer.balance - this.customer.widthDrawAmount;

Console.WriteLine("Balance after Withdrawal : :" + this.customer.balance.ToString());

//return "Balance is :" + this.customer.balance.ToString();

}

else

{

Console.WriteLine("InSufficient balance to widthdraw and your balance is : " + this.customer.balance.ToString());

//return "InSufficient balance to widthdraw and your balance is : " + this.customer.balance.ToString();

}

}

finally

{

Monitor.Exit(this);

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\* Transaction Finished \*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ");

}

}

public void Deposit( )

{

this.customer.balance = this.customer.balance + this.customer.widthDrawAmount;

Console.WriteLine("Balance is :" + this.customer.balance.ToString());

//return "Balance is :" + this.customer.balance.ToString();

}

public string showBalance()

{

Console.WriteLine("Final Balance Amount is :" + this.customer.balance.ToString());

return "Final Balance Amount is :" + this.customer.balance.ToString();

}

}

public class Customer

{

public string accountNumber { get; set; }

public string customerName { get; set; }

public string jointCustomerName { get; set; }

public string customerAddress { get; set; }

public double balance { get; set; }

public double widthDrawAmount { get; set; }

}

}

