Code:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

namespace CSharp\_Assignment\_3

{

class Program

{

static void Main(string[] args)

{

Account account = new Account() { account\_number = "AB12345362346", customer\_name = "Subasree", customer\_address = "XYZ", balance = 200000 };

double Withdraw = 2000;

double deposit = 1000;

Bank bk = new Bank(account, Withdraw, deposit);

Thread[] thread = new Thread[2];

Console.WriteLine("-------------Withdraw thread---------");

thread[0] =new Thread( new ThreadStart(bk.WithDraw));

thread[0].Name = "Thread 0 ";

Console.WriteLine("-------------Deposit thread---------");

thread[1] = new Thread(new ThreadStart(bk.Deposit));

thread[1].Name = "Thread 1 ";

foreach (Thread t in thread)

{

t.Start();

}

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CSharp\_Assignment\_3

{

public class Account

{

public string account\_number { get; set;}

public string customer\_name { get; set; }

public string customer\_address { get; set; }

public double balance { get; set; }

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

namespace CSharp\_Assignment\_3

{

public class Bank

{

#region Constructor

public Bank()

{

}

public Bank(Account Account,double WithDrawAmount,double Deposit)

{

this.account = Account;

this.WithDrawAmount = WithDrawAmount;

this.deposit = Deposit;

}

#endregion Constructor

public Account account { get; set; }

public double WithDrawAmount { get; set; }

public double deposit { get; set; }

#region Method

public void WithDraw()

{

Monitor.Enter(this);

Console.WriteLine("----------------WithDraw Starts-------------------");

if(this.account.balance > WithDrawAmount )

{

Thread.Sleep(1000);

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

Console.WriteLine("Amount to be withdraw : "+ WithDrawAmount);

Console.WriteLine("Amount before withdraw : "+ this.account.balance);

this.account.balance = this.account.balance - WithDrawAmount;

Console.WriteLine("Amount After withdraw : "+ this.account.balance);

}

else

{

Console.WriteLine("Insufficent Balance in account {0} with Amount {1}", this.account.account\_number, this.account.balance);

}

Monitor.Exit(this);

Console.WriteLine("----------------WithDraw Ends-------------------");

}

public void Deposit()

{

Monitor.Enter(this);

Console.WriteLine("----------------Deposit Starts-------------------");

if (this.deposit > 0)

{

Thread.Sleep(1000);

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

Console.WriteLine("Amount to be deposit : "+ deposit);

Console.WriteLine("Amount before deposit : "+ this.account.balance);

this.account.balance = this.account.balance + deposit ;

Console.WriteLine("Amount After deposit : "+ this.account.balance);

}

else

{

Console.WriteLine("No amount {1} to deposit in account {0} ", this.account.account\_number, this.account.balance);

}

Monitor.Exit(this);

Console.WriteLine("----------------Deposit Ends-------------------");

}

#endregion Method

}

}

Output :

