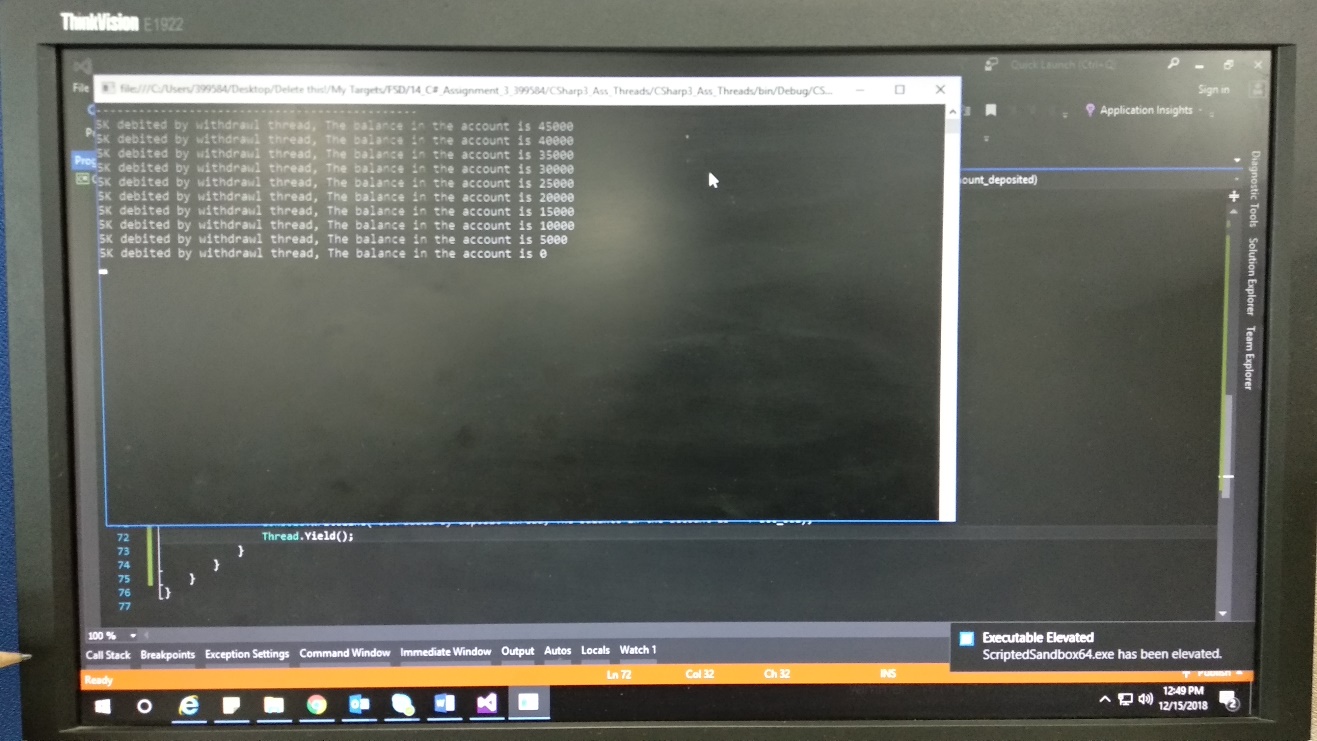
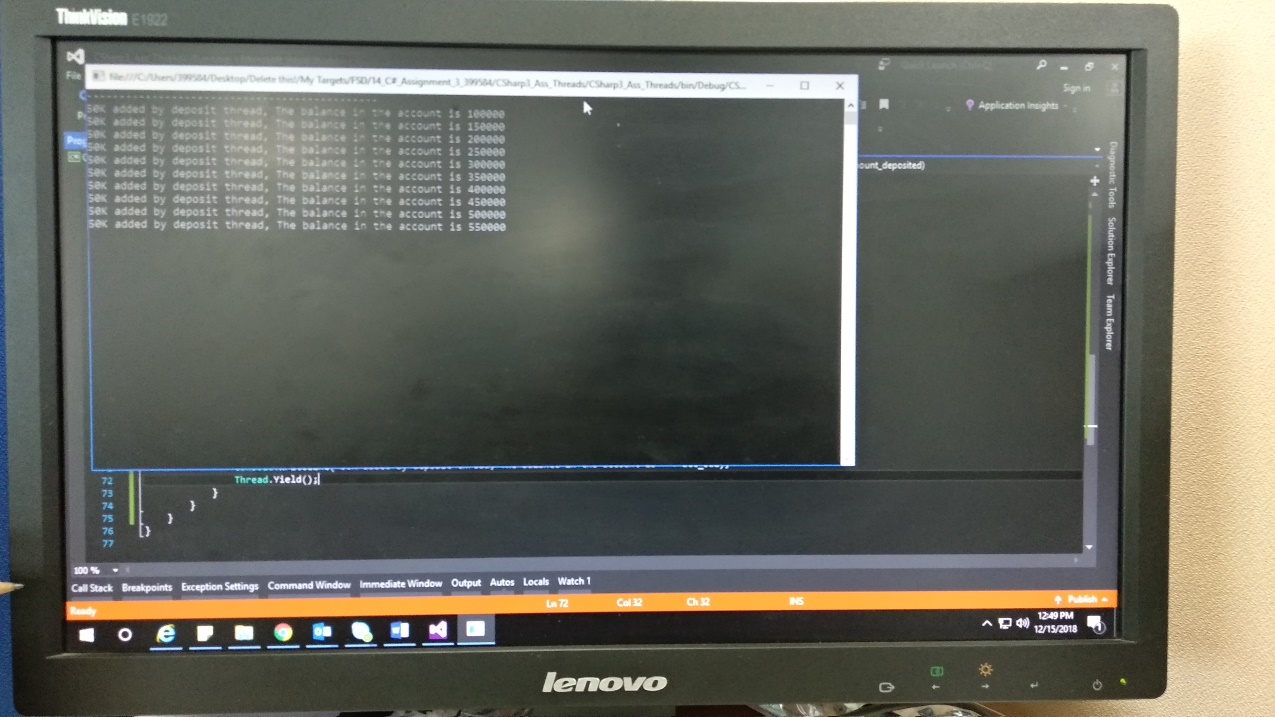
Consider a banking application in which there is an entity called Account. Its attributes are account number, customer name, customer’s address and balance. It provides the functionality of withdrawal and deposit. There can be two users for a particular bank account (joint account). The two users can perform any type of transaction on the account. Construct a code that performs deposit and withdrawal of money from the same account by two users concurrently. The balance printed after the transactions should be correct.

Hint: Use Monitor class to achieve the same.

**Output Screenshots:**

****

****

**Code:**

using System;

using System.Threading;

namespace CSharp3\_Ass\_Threads

{

public class Program

{

public static void Main(string[] args)

{

Account account = new Account(1234, "Balaji Singh", "Kurnool", 50000);

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

//account.withdrawl(6678.75);

// Two ways to create threads in C#

//ThreadStart threadstart\_withdrawl = new ThreadStart(account.withdrawl(6678.75));

//ThreadStart threadstart\_deposit = new ThreadStart(account.deposit(50000));

//Thread thread\_withdrawl = new Thread(threadstart\_withdrawl);

//Thread thread\_deposit = new Thread(threadstart\_deposit);

//Thread thread\_withdrawl = new Thread(new ThreadStart(account.withdrawl));

//Thread thread\_deposit = new Thread(new ThreadStart(account.deposit));

// Both ways are correct

Thread thread\_withdrawl = new Thread(() => account.withdrawl(5000));

Thread thread\_deposit = new Thread(() => account.deposit(50000));

thread\_withdrawl.Start();

thread\_deposit.Start();

thread\_withdrawl.Join();

thread\_deposit.Join();

Console.ReadLine();

}

}

public class Account

{

static object locker = new object();

public int acc\_no { get; set; }

public string cust\_name { get; set; }

public string cust\_address { get; set; }

public double acc\_bal { get; set; }

public Account()

{

}

public Account(int ac\_number, string customer\_nm, string cust\_add, double ac\_balance)

{

acc\_no = ac\_number;

cust\_name = customer\_nm;

cust\_address = cust\_add;

acc\_bal = ac\_balance;

}

public void withdrawl(double amount\_withdrawn)

{

for (int i = 0; i < 10; i++)

{

Monitor.Enter(locker);

acc\_bal = acc\_bal - amount\_withdrawn;

Console.WriteLine("5K debited by withdrawl thread, The balance in the account is " + acc\_bal);

Thread.Yield();

}

}

public void deposit(double amount\_deposited)

{

for (int i = 0; i < 10; i++)

{

Monitor.Enter(locker);

acc\_bal = acc\_bal + amount\_deposited;

Console.WriteLine("50K added by deposit thread, The balance in the account is " + acc\_bal);

Thread.Yield();

}

}

}

}