14b.Assume that two brothers, Joe and John, share a common bank account. They both can, independently, read the balance, make a deposit, and withdraw some money. Implement java application demonstrate how the transaction in a bank can be carried out concurrently.

**Code:-**

**class BankAccount {**

**private double balance;**

**public BankAccount(double initialBalance) {**

**this.balance = initialBalance;**

**}**

**public synchronized double getBalance() {**

**return balance;**

**}**

**public synchronized void deposit(double amount) {**

**balance += amount;**

**System.out.println("Deposited: $" + amount);**

**}**

**public synchronized void withdraw(double amount) {**

**if (balance >= amount) {**

**balance -= amount;**

**System.out.println("Withdrawn: $" + amount);**

**} else {**

**System.out.println("Insufficient balance for withdrawal.");**

**}**

**}**

**}**

**class Joe implements Runnable {**

**private BankAccount account;**

**public Joe(BankAccount account) {**

**this.account = account;**

**}**

**@Override**

**public void run() {**

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

**double depositAmount = Math.random() \* 100;**

**account.deposit(depositAmount);**

**}**

**}**

**}**

**class John implements Runnable {**

**private BankAccount account;**

**public John(BankAccount account) {**

**this.account = account;**

**}**

**@Override**

**public void run() {**

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

**double withdrawAmount = Math.random() \* 50;**

**account.withdraw(withdrawAmount);**

**}**

**}**

**}**

**public class BankDemo {**

**public static void main(String[] args) {**

**BankAccount sharedAccount = new BankAccount(1000);**

**Thread joeThread = new Thread(new Joe(sharedAccount));**

**Thread johnThread = new Thread(new John(sharedAccount));**

**joeThread.start();**

**johnThread.start();**

**try {**

**joeThread.join();**

**johnThread.join();**

**} catch (InterruptedException e) {**

**e.printStackTrace();**

**}**

**System.out.println("Final Balance: $" + sharedAccount.getBalance());**

**}**

**}**