Online Banking System

Approach: Incorporating synchronization with multithreading.

```
class Bank {

    static int total = 100;
    static synchronized void withdrawn(String name,
                                       int withdrawal)
    {
        if (total >= withdrawal) {
            System.out.println(name + " withdrawn "
                               + withdrawal);
            total = total - withdrawal;
            System.out.println("Balance after withdrawal: "
                               + total);
            try {
                Thread.sleep(1000);
            }
            catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
         else {
            System.out.println(name
                               + " you can not withdraw "
                               + withdrawal);
            System.out.println("your balance is: " + total);

            try {

                Thread.sleep(1000);
            }

            catch (InterruptedException e) {

                e.printStackTrace();
            }
        }
    }

    static synchronized void deposit(String name,
                                     int deposit)
    {
        System.out.println(name + " deposited " + deposit);
```

```
        total = total + deposit;
        System.out.println("Balance after deposit: "
                           + total);

        try {


            Thread.sleep(1000);
        }

        catch (InterruptedException e) {

            e.printStackTrace();
        }
    }
}

class ThreadWithdrawal extends Thread {

    Bank object;
    String name;
    int dollar;

    ThreadWithdrawal(Bank ob, String name, int money)
    {
        this.object = ob;
        this.name = name;
        this.dollar = money;
    }

    public void run() { object.withdrawn(name, dollar); }
}

class ThreadDeposit extends Thread {

    Bank object;
    String name;
    int dollar;

    ThreadDeposit(Bank ob, String name, int money)
    {
        this.object = ob;
        this.name = name;
        this.dollar = money;
    }

    public void run() { object.deposit(name, dollar); }
}
```

```
class GFG {

    public static void main(String[] args)
    {



        Bank obj = new Bank();

        ThreadWithdrawal t1
            = new ThreadWithdrawal(obj, "Arnab", 20);
        ThreadWithdrawal t2
            = new ThreadWithdrawal(obj, "Monodwip", 40);
        ThreadDeposit t3
            = new ThreadDeposit(obj, "Mukta", 35);
        ThreadWithdrawal t4
            = new ThreadWithdrawal(obj, "Rinkel", 80);
        ThreadWithdrawal t5
            = new ThreadWithdrawal(obj, "Shubham", 40);

        t1.start();
        t2.start();
        t3.start();
        t4.start();
        t5.start();
    }
}
```

Output:

Arnab withdrawn 20

Balance after withdrawal: 80

Shubham withdrawn 40

Balance after withdrawal: 40

Rinkel you can not withdraw 80

your balance is: 40

Mukta deposited 35

Balance after deposit: 75

Monodwip withdrawn 40

Balance after withdrawal: 35