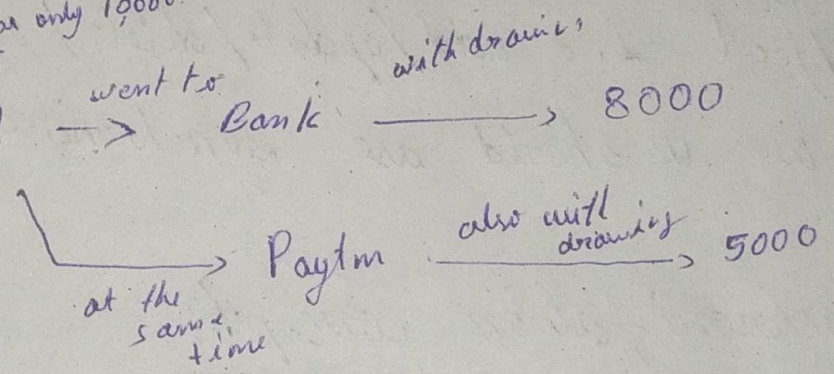
Race condition:-

A race-condition is a condition in which the critical section (a part of the program where shared memory is accessed) is concurrently executed by two or more threads. It leads to incorrect behaviour of a program.

in simple terms

a condition in which two or more threads compete together to get certain shared resources.



So, to solve the data inconsistency problem in java **synchronized** keyword is used. So the thread-safety is achieved and race condition is avoided by the help of "synchronized" keyword

class Demo {

```
    common c1 = new common();
    common c2 = new common();
Thread t1 = new thread (c1, "Ram");
Thread t2 = new Thread(c2, "Shyam");
    t1.start();
    t2.start();
```

```
Public class Common {

    Public synchronized void fun1(String name)
        Syso ("welcome");
    try{
        Thread.sleep(1000);
    }
    catch (Exception ee){
    }
    Syso (name);
    }
}
```

→ Whenever we need to suspend a synchronized thread unconditionally then we use

wait() method

→ Whenever we need to resume a suspended thread then we use

notify() method

This is known as thread-synchronized or inter-thread communication.

Note:- We can call wait(), notify, notifyAll() only in the synchronized block or synchronized methods. otherwise we will get Runtime exception

To call wait() or notify method on any object we must have that particular object lock, otherwise we will get a runtime exception called IllegalMonitorStateException