package com.mkyong;

import java.util.concurrent.Semaphore;

public class SemaphoreTest {

// max 4 people

static Semaphore semaphore = new Semaphore(4);

static class MyATMThread extends Thread {

String name = "";

MyATMThread(String name) {

this.name = name;

}

public void run() {

try {

System.out.println(name + " : acquiring lock...");

System.out.println(name + " : available Semaphore permits now: "

+ semaphore.availablePermits());

semaphore.acquire();

System.out.println(name + " : got the permit!");

try {

for (int i = 1; i <= 5; i++) {

System.out.println(name + " : is performing operation " + i

+ ", available Semaphore permits : "

+ semaphore.availablePermits());

// sleep 1 second

Thread.sleep(1000);

}

} finally {

// calling release() after a successful acquire()

System.out.println(name + " : releasing lock...");

semaphore.release();

System.out.println(name + " : available Semaphore permits now: "

+ semaphore.availablePermits());

}

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

public static void main(String[] args) {

System.out.println("Total available Semaphore permits : "

+ semaphore.availablePermits());

MyATMThread t1 = new MyATMThread("A");

t1.start();

MyATMThread t2 = new MyATMThread("B");

t2.start();

MyATMThread t3 = new MyATMThread("C");

t3.start();

MyATMThread t4 = new MyATMThread("D");

t4.start();

MyATMThread t5 = new MyATMThread("E");

t5.start();

MyATMThread t6 = new MyATMThread("F");

t6.start();

}}