import java.util.concurrent.atomic.AtomicBoolean;

import java.util.concurrent.atomic.AtomicInteger;

class TokenRingMutex extends Thread {

private static final int NUM\_PROCESSES = 5;

private static final AtomicInteger tokenHolder = new AtomicInteger(0);

private static final AtomicBoolean token = new AtomicBoolean(true);

private int id;

public TokenRingMutex(int id) {

this.id = id;

}

private void criticalSection() {

System.out.println("Process " + id + " is in the critical section.");

try {

Thread.sleep(1000); // Simulate work

} catch (InterruptedException e) {

e.printStackTrace();

}

System.out.println("Process " + id + " is leaving the critical section.");

}

@Override

public void run() {

while (true) {

if (tokenHolder.get() == id) {

criticalSection();

passToken();

}

}

}

private void passToken() {

int next = (id + 1) % NUM\_PROCESSES;

System.out.println("Process " + id + " passes token to Process " + next);

tokenHolder.set(next);

}

public static void main(String[] args) {

TokenRingMutex[] processes = new TokenRingMutex[NUM\_PROCESSES];

for (int i = 0; i < NUM\_PROCESSES; i++) {

processes[i] = new TokenRingMutex(i);

processes[i].start();

}

}

}

Output:

javac TokenRIngMutex.java

java TokenRingMutex

Process 0 is in the critical section.

Process 0 is leaving the critical section.

Process 0 passes token to Process 1

Process 1 is in the critical section.

Process 1 is leaving the critical section.

Process 1 passes token to Process 2

Process 2 is in the critical section.

Process 2 is leaving the critical section.

Process 2 passes token to Process 3

Process 3 is in the critical section.

Process 3 is leaving the critical section.

Process 3 passes token to Process 4

Process 4 is in the critical section.

Process 4 is leaving the critical section.

Process 4 passes token to Process 0

Process 0 is in the critical section.

Process 0 is leaving the critical section.

Process 0 passes token to Process 1