**Experiment 04: Mutual Exclusion using centralized algorithm**

import java.io.\*;

import java.util.\*;

class mutex1

{

public static void main(String ag[])

{

//mutex m = new mutex();

Queue wait = new LinkedList();

//int wait[] = new int[4];

int cs=0,flag = 0;

int i,j=5,k,removedP;

int process[] = new int[5];

process[0] = 10;

process[1] = 20;

process[2] = 30;

process[3] = 40;

process[4] = 50;

System.out.println("Process NO: 50 is the coordinator\n");

Random r = new Random();

for(i=0;i<process.length;i++)

{

System.out.println("p: "+process[i]);

}

//while(j!=0)

//{

for(i=0;i<4;i++)

{

//int rnd = r.nextInt(5);

System.out.println(process[i]+" is requesting");

if(process[i]!=50)

{

if(flag == 0 && wait.isEmpty())

{

cs = process[i];

System.out.println("Critical Section: "+cs);

flag = 1;

}

else

if(flag == 1 && wait.size()<2)

{

System.out.println("Waiting Queue: ");

wait.add(process[i]);

}

else

if(wait.size()==2 && flag == 1)

{

cs = 0;

System.out.println("PROCESS HAS LEFT..CRITICAL SECTION IS AVAILABLE");

//removedP = (int)wait.remove();

System.out.println(wait.remove()+"granted CS");

wait.add(process[i]);

}

System.out.println(wait);

}

else

{

System.out.println("I am the Coordinator");

}

}

}

}

Output

