Ministerul Educaţiei, al Culturii și Cercetării al Republicii Moldova

Universitatea Tehnică a Moldovei

Departamentul Informatică și Ingineria Sistemelor

**RAPORT**

Lucrarea de laborator nr.1

ASO

A efectuat:

st. gr. C-171 D. Melniciuc

A verificat:

dr., conf.univ. L.Rotaru

Chişinău 2020

Varianta:4

X Y Z D Tip Obiecte

3 2 12 11 Consoane

Listingul programului:

// package com.company;

// X Y Z D Tip Obiecte

// 3 2 12 11 Consoane

// X prod

// Y consum

// Z nr obj

// D depo

import java.util.\*;

public class Main {

int X = 3;

int Y = 2;

int Z = 44;

int D = 11;

int[] state = new int[Y];

int countC = 0;

int countP = 0;

int nrP = 0;

int nrC = 0;

public enum consonants {

q, w, r, t, y, p, s, d, f, g, h, k, l, z, x, c, v, b, n, m

}

private static char extract() {

int pick = new Random().nextInt(consonants.values().length);

return consonants.values()[pick].toString().charAt(0);

}

public class Sync {

Stack stack = new Stack();

private boolean put = true;

private boolean get = false;

private boolean full() {

if (stack.size() == D ) {

// System.out.println("plin");

return false;

} else

return true;

}

private boolean empty() {

if (stack.size() < 1) {

// System.out.println("gol");

return false;

} else

return true;

}

public synchronized char get() {

get = empty();

while (!get) {

try { wait(); }

catch (InterruptedException e) {

get = empty();

}

}

if (get) {

try {

char temp = this.stack.pop().toString().charAt(0);

notifyAll();

put = this.full();

return temp;

} catch (EmptyStackException e) {}

}

return '0';

}

public synchronized void put(char ch) {

put = full();

while (!put) {

try { wait(); }

catch (InterruptedException e) {

put = full();

}

}

if (put) {

this.stack.push(ch);

notifyAll();

get = this.empty();

}

}

}

public int countP() {

nrP++;

return nrP;

}

public int countC() {

nrC++;

return nrC;

}

private boolean checkP() {

countP++;

// System.out.println("countP = " + countP);

if ( countP == Z\*2 )

return false;

else

return true;

}

private boolean checkC() {

countC++;

// System.out.println("\tcountC = " + countC);

if ( countC == Z\*2 )

return false;

else

return true;

}

public class Producer extends Thread {

private Sync buffer;

private int number;

private boolean p;

private int nr = 0;

public Producer(Sync c, int number) {

buffer = c;

this.number = number;

}

@Override

public void run() {

while (true) {

p = checkP();

if ( p == false ) {

System.out.println("prod == Z");

try { sleep(100000); }

catch (InterruptedException e) {}

}

char temp = extract();

System.out.println(countP() + ") Producer " + this.number + " puts: " + temp);

buffer.put(temp);

try { sleep((int) (Math.random() \* 100)); }

catch (InterruptedException e) {}

}

}

}

public class Consumer extends Thread {

private Sync buffer;

private int number;

private int max = 3;

private int consonants\_Used = 0;

private boolean c;

int nr = 0;

public Consumer(Sync c, int number) {

buffer = c;

this.number = number;

}

@Override

public void run() {

char ch;

//

// while (consonants\_Used < max) {

while (true) {

c = checkC();

ch = buffer.get();

if (ch != '0') {

consonants\_Used++;

System.out.println("\t" + countC() + ") Consumer " + this.number + " got: " + ch + ", used " + consonants\_Used + " letters");

}

if ( c == false ) {

// System.out.println("\tconsum == Z; c = " + c);

System.exit(0);

try { sleep(100000); }

catch (InterruptedException e) {}

}

}

// state[this.number - 1] = 1;

// if (state()) { System.exit(0); }

}

}

private boolean state() {

for (int i = 0; i < Y; i++) {

if (state[i] == 0)

return false;

}

return true;

}

public void exe() {

Sync c = new Sync();

Producer p1 = new Producer(c, 1);

Producer p2 = new Producer(c, 2);

Producer p3 = new Producer(c, 3);

Producer p4 = new Producer(c, 4);

Consumer c1 = new Consumer(c, 1);

Consumer c2 = new Consumer(c, 2);

Consumer c3 = new Consumer(c, 3);

p1.start();

p2.start();

p3.start();

p4.start();

c1.start();

c2.start();

c3.start();

}

public static void main(String[] args) {

try {

new ProcessBuilder("cmd", "/c", "cls").inheritIO().start().waitFor();

} catch (Exception e1) {

System.out.println(e1);

}

new Main().exe();

}

}

Output:

1) Producer 1 puts: b

2) Producer 2 puts: s

4) Producer 4 puts: l

3) Producer 3 puts: f

3) Consumer 2 got: l, used 1 letters

2) Consumer 3 got: s, used 1 letters

4) Consumer 2 got: f, used 2 letters

5) Producer 1 puts: m

1) Consumer 1 got: b, used 1 letters

5) Consumer 3 got: m, used 2 letters

6) Producer 1 puts: s

6) Consumer 2 got: s, used 3 letters

7) Producer 4 puts: n

7) Consumer 3 got: n, used 3 letters

8) Producer 1 puts: s

8) Consumer 2 got: s, used 4 letters

9) Producer 3 puts: f

9) Consumer 3 got: f, used 4 letters

10) Producer 1 puts: h

10) Consumer 2 got: h, used 5 letters

11) Producer 2 puts: z

11) Consumer 3 got: z, used 5 letters

12) Producer 3 puts: q

13) Producer 4 puts: p

12) Consumer 2 got: q, used 6 letters

13) Consumer 3 got: p, used 6 letters

14) Producer 3 puts: y

14) Consumer 1 got: y, used 2 letters

15) Producer 1 puts: w

15) Consumer 3 got: w, used 7 letters

16) Producer 1 puts: r

16) Consumer 1 got: r, used 3 letters

17) Producer 4 puts: s

17) Consumer 3 got: s, used 8 letters

18) Producer 1 puts: n

18) Consumer 1 got: n, used 4 letters

19) Producer 2 puts: k

19) Consumer 3 got: k, used 9 letters

20) Producer 2 puts: v

20) Consumer 1 got: v, used 5 letters

21) Producer 3 puts: c

21) Consumer 3 got: c, used 10 letters

23) Producer 1 puts: b

22) Producer 4 puts: l

22) Consumer 1 got: b, used 6 letters

23) Consumer 3 got: l, used 11 letters

24) Producer 4 puts: q

24) Consumer 2 got: q, used 7 letters

25) Producer 1 puts: s

25) Consumer 3 got: s, used 12 letters

26) Producer 2 puts: d

26) Consumer 2 got: d, used 8 letters

27) Producer 2 puts: c

27) Consumer 3 got: c, used 13 letters

28) Producer 3 puts: z

29) Producer 4 puts: q

28) Consumer 2 got: z, used 9 letters

29) Consumer 3 got: q, used 14 letters

30) Producer 2 puts: w

30) Consumer 1 got: w, used 7 letters

31) Producer 1 puts: c

31) Consumer 3 got: c, used 15 letters

32) Producer 1 puts: d

32) Consumer 1 got: d, used 8 letters

33) Producer 3 puts: f

33) Consumer 3 got: f, used 16 letters

34) Producer 4 puts: s

34) Consumer 1 got: s, used 9 letters

35) Producer 2 puts: k

35) Consumer 3 got: k, used 17 letters

36) Producer 4 puts: k

36) Consumer 1 got: k, used 10 letters

37) Producer 1 puts: d

37) Consumer 3 got: d, used 18 letters

38) Producer 3 puts: d

39) Producer 2 puts: m

38) Consumer 1 got: d, used 11 letters

39) Consumer 3 got: m, used 19 letters

40) Producer 2 puts: y

40) Consumer 2 got: y, used 10 letters

41) Producer 4 puts: y

41) Consumer 3 got: y, used 20 letters

42) Producer 2 puts: k

42) Consumer 2 got: k, used 11 letters

43) Producer 3 puts: t

43) Consumer 3 got: t, used 21 letters

44) Producer 1 puts: l

44) Consumer 2 got: l, used 12 letters

45) Producer 4 puts: l

45) Consumer 3 got: l, used 22 letters

C:\code\universitate\aso>