Ministerul Educaţiei, al Culturii și Cercetării al Republicii Moldova

Universitatea Tehnică a Moldovei

Departamentul Informatică și Ingineria Sistemelor

**RAPORT**

Lucrarea de laborator nr.3

ASO

A efectuat:

st. gr. C-171 D. Melniciuc

A verificat:

dr., conf.univ. L.Rotaru

Chişinău 2020

Varianta:4

4.

// X Y Z D Tip Obiecte

// 3 2 49 7 Consoane

Listingul programului:

import java.util.LinkedList;

import java.util.concurrent.Semaphore;

import java.util.concurrent.BrokenBarrierException;

import java.util.concurrent.CyclicBarrier;

import java.util.Random;

import java.io.FileReader;

import java.io.IOException;

import java.io.BufferedReader;

import java.util.Scanner;

import java.nio.charset.StandardCharsets;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;

public class lab3\_bin {

static Object LOCK = new Object();

static LinkedList list = new LinkedList();

static Semaphore sem = new Semaphore(0);

static Semaphore mutex = new Semaphore(1);

static CyclicBarrier barrier = new CyclicBarrier(2);

static int await = 0;

static boolean bl = false;

static String[] consonants = {"q", "w", "r", "t", "y", "p", "s", "d", "f", "g", "h", "k", "l", "z", "x", "c", "v", "b", "n", "m"};

static int cntP = 0;

static public int countP () {

// System.out.println("cntP " + cntP);

return cntP++;

}

static public synchronized boolean getP () {

// System.out.println("cntP " + cntP);

if (cntP >= 48 + 1 ) return true;

else return false;

}

static public synchronized String send (String b) {

String z = "false";

if (b == "send")

z = "true";

else

z = "false";

if (b == "go")

return z;

return "w";

}

static class Consumer extends Thread {

String name;

private CyclicBarrier barrier;

public Consumer (CyclicBarrier barrier, String name) {

this.name = name;

this.barrier = barrier;

}

public void run() {

try {

while (true) {

if ( getP() == true ) {

System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~");

// Thread.sleep(200);

System.exit(1);

}

Thread.sleep(100);

if (list.size() == 7) {

await = barrier.await();

sem.acquire(1);

mutex.acquire();

while (list.size() != 0) {

Random rand = new Random();

int z = rand.nextInt(list.size());

System.out.print("\tConsumer " + name + " got: " + list.get(z) );

list.remove(z);

System.out.println(" | depo " + list.size());

System.out.println("\t" + list);

if (bl == false)

Thread.sleep(100\*100\*list.size());

}

mutex.release();

send("send");

}

}

} catch (Exception x) {

x.printStackTrace();

}

}

}

static class Producer extends Thread {

private CyclicBarrier barrier;

String name;

public Producer (String name) {

this.name = name;

}

// public Producer (CyclicBarrier barrier, String name) {

// this.name = name;

// this.barrier = barrier;

// }

public void run() {

try {

Random rd = new Random();

int rand = 0;

int k = 0;

while (true) {

mutex.acquire();

rand = rd.nextInt(consonants.length);

// int await = barrier.await();

if ( list.size() < 7 || send("go") == "true") {

k = countP() + 1;

list.add(consonants[rand]);

// System.out.print(k + ") Producer " + name + " put: " + consonants[rand]);

System.out.print("Producer " + name + " put: " + consonants[rand]);

System.out.println(" | depo " + list.size());

System.out.println(list);

send("stop");

}

// if (await == 0) {

// System.out.println("all prod. put someting");

// }

mutex.release();

sem.release(1);

Thread.sleep(200);

}

} catch (Exception x) {

x.printStackTrace();

}

}

}

public static void main(String [] args) throws NoSuchAlgorithmException {

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

catch (Exception e1) { System.out.println(e1); }

String pass = "";

BufferedReader objReader = null;

try {

String strCurrentLine;

objReader = new BufferedReader(new FileReader("pass.txt"));

while ((strCurrentLine = objReader.readLine()) != null) {

pass += strCurrentLine;

}

} catch (IOException e) { e.printStackTrace(); } finally {

try {

if (objReader != null)

objReader.close();

} catch (IOException ex) { ex.printStackTrace(); }

}

MessageDigest md = MessageDigest.getInstance("MD5");

byte[] hashInBytes = md.digest(pass.getBytes(StandardCharsets.UTF\_8));

StringBuilder sb = new StringBuilder();

for (byte b : hashInBytes) {

sb.append(String.format("%02x", b));

}

String pass2 = sb.toString();

String pass2check = "23ae8fbbe680618e3abbf35d85d716c1";

if ( pass2.equals(pass2check) ) {

bl = true;

System.out.println("\t[+] Authorized access");

} else {

System.out.println("Unauthorized copy\nPassword file not Found\nPlease contact Dima with a kebab");

System.out.println("Press Any Key To Continue...");

new java.util.Scanner(System.in).nextLine();

System.exit(1);

}

System.out.println(consonants.length);

CyclicBarrier barrier = new CyclicBarrier(2);

CyclicBarrier barrier2 = new CyclicBarrier(3);

new Producer("1").start();

new Producer("2").start();

new Producer("3").start();

new Consumer(barrier,"1").start();

new Consumer(barrier,"2").start();

}

}

Output:

20

Producer 2 put: p | depo 1

[p]

Producer 3 put: y | depo 2

[p, y]

Producer 1 put: q | depo 3

[p, y, q]

Producer 3 put: f | depo 4

[p, y, q, f]

Producer 1 put: l | depo 5

[p, y, q, f, l]

Producer 2 put: d | depo 6

[p, y, q, f, l, d]

Producer 3 put: p | depo 7

[p, y, q, f, l, d, p]

Consumer 2 got: p | depo 6

[p, y, q, f, l, d]

Consumer 2 got: l | depo 5

[p, y, q, f, d]

Consumer 2 got: y | depo 4

[p, q, f, d]

Consumer 2 got: d | depo 3

[p, q, f]

Consumer 2 got: q | depo 2

[p, f]

Consumer 2 got: p | depo 1

[f]

Consumer 2 got: f | depo 0

[]

Producer 3 put: c | depo 1

[c]

Producer 1 put: w | depo 2

[c, w]

Producer 2 put: s | depo 3

[c, w, s]

Producer 3 put: m | depo 4

[c, w, s, m]

Producer 1 put: h | depo 5

[c, w, s, m, h]

Producer 2 put: v | depo 6

[c, w, s, m, h, v]

Producer 1 put: d | depo 7

[c, w, s, m, h, v, d]

Consumer 2 got: v | depo 6

[c, w, s, m, h, d]

Consumer 2 got: s | depo 5

[c, w, m, h, d]

Consumer 2 got: d | depo 4

[c, w, m, h]

Consumer 2 got: h | depo 3

[c, w, m]

Consumer 2 got: m | depo 2

[c, w]

Consumer 2 got: c | depo 1

[w]

Consumer 2 got: w | depo 0

[]

Producer 3 put: c | depo 1

[c]

Producer 1 put: f | depo 2

[c, f]

Producer 2 put: k | depo 3

[c, f, k]

Producer 3 put: h | depo 4

[c, f, k, h]

Producer 1 put: l | depo 5

[c, f, k, h, l]

Producer 2 put: s | depo 6

[c, f, k, h, l, s]

Producer 3 put: f | depo 7

[c, f, k, h, l, s, f]

Consumer 1 got: l | depo 6

[c, f, k, h, s, f]

Consumer 1 got: f | depo 5

[c, f, k, h, s]

Consumer 1 got: f | depo 4

[c, k, h, s]

Consumer 1 got: h | depo 3

[c, k, s]

Consumer 1 got: c | depo 2

[k, s]

Consumer 1 got: k | depo 1

[s]

Consumer 1 got: s | depo 0

[]

Producer 1 put: n | depo 1

[n]

Producer 3 put: c | depo 2

[n, c]

Producer 2 put: d | depo 3

[n, c, d]

Producer 1 put: h | depo 4

[n, c, d, h]

Producer 3 put: v | depo 5

[n, c, d, h, v]

Producer 2 put: y | depo 6

[n, c, d, h, v, y]

Producer 1 put: s | depo 7

[n, c, d, h, v, y, s]

Consumer 1 got: c | depo 6

[n, d, h, v, y, s]

Consumer 1 got: y | depo 5

[n, d, h, v, s]

Consumer 1 got: d | depo 4

[n, h, v, s]

Consumer 1 got: h | depo 3

[n, v, s]

Consumer 1 got: n | depo 2

[v, s]

Consumer 1 got: v | depo 1

[s]

Consumer 1 got: s | depo 0

[]

Producer 3 put: v | depo 1

[v]

Producer 1 put: h | depo 2

[v, h]

Producer 2 put: d | depo 3

[v, h, d]

Producer 3 put: b | depo 4

[v, h, d, b]

Producer 1 put: p | depo 5

[v, h, d, b, p]

Producer 2 put: m | depo 6

[v, h, d, b, p, m]

Producer 3 put: p | depo 7

[v, h, d, b, p, m, p]

Consumer 2 got: m | depo 6

[v, h, d, b, p, p]

Consumer 2 got: d | depo 5

[v, h, b, p, p]

Consumer 2 got: v | depo 4

[h, b, p, p]

Consumer 2 got: b | depo 3

[h, p, p]

Consumer 2 got: h | depo 2

[p, p]

Consumer 2 got: p | depo 1

[p]

Consumer 2 got: p | depo 0

[]

Producer 1 put: z | depo 1

[z]

Producer 3 put: y | depo 2

[z, y]

Producer 2 put: x | depo 3

[z, y, x]

Producer 1 put: n | depo 4

[z, y, x, n]

Producer 2 put: c | depo 5

[z, y, x, n, c]

Producer 3 put: t | depo 6

[z, y, x, n, c, t]

Producer 1 put: m | depo 7

[z, y, x, n, c, t, m]

Consumer 2 got: t | depo 6

[z, y, x, n, c, m]

Consumer 2 got: n | depo 5

[z, y, x, c, m]

Consumer 2 got: m | depo 4

[z, y, x, c]

Consumer 2 got: c | depo 3

[z, y, x]

Consumer 2 got: z | depo 2

[y, x]

Consumer 2 got: y | depo 1

[x]

Consumer 2 got: x | depo 0

[]

Producer 1 put: t | depo 1

[t]

Producer 2 put: q | depo 2

[t, q]

Producer 3 put: l | depo 3

[t, q, l]

Producer 1 put: n | depo 4

[t, q, l, n]

Producer 3 put: y | depo 5

[t, q, l, n, y]

Producer 2 put: m | depo 6

[t, q, l, n, y, m]

Producer 1 put: l | depo 7

[t, q, l, n, y, m, l]

Consumer 1 got: m | depo 6

[t, q, l, n, y, l]

Consumer 1 got: l | depo 5

[t, q, l, n, y]

Consumer 1 got: l | depo 4

[t, q, n, y]

Consumer 1 got: y | depo 3

[t, q, n]

Consumer 1 got: n | depo 2

[t, q]

Consumer 1 got: t | depo 1

[q]

Consumer 1 got: q | depo 0

[]

~~~~~~~~~~~~~~~~~~~~~~~~~~

~~~~~~~~~~~~~~~~~~~~~~~~~~

C:\code\universitate\aso>