UNA-PUNO
F I M E E S
E. P. DE INGENIERÍA DE SISTEMAS

TAREA 1

Tarea: Productor-Consumidor

CURSO:
SISTEMAS OPERATIVOS
DOCENTE:
ING. FERNANDEZ CHAMBI MAYENKA

AUTOR:
GUTIERREZ CHAMBILLA RUSSO WILLIAMS
russodx@gmail.com
951 020 703

2023
SEMESTRE I

TAREA 01

Objetivo:

Programar el problema del “productor - consumidor” para concretar la idea de los procesos colaborativos.

Indicaciones:

1. Programe en el lenguaje de programación de su elección el problema del “Productor-Consumidor” con Memoria Compartida.
2. Explique y muestre el funcionamiento de su algoritmo en un video.
3. Comparta el enlace del video y del programa en una plataforma de código (Github) en un archivo PDF que incluye los datos de la tarea, integrantes.

1. REFERENCIA

```
package com.mycompany.soproductor_consumidor;
import java.util.concurrent.Semaphore;
class Q {

    int item;
    static Semaphore semCon = new Semaphore(0);
    static Semaphore semProd = new Semaphore(1);
    void get()
    {
        try {
            semCon.acquire();
        }
        catch (InterruptedException e) {
            System.out.println("InterruptedException caught");
        }

        System.out.println("Consumer consumed item : " + item);
        semProd.release();
    }
    void put(int item)
    {
        try {
            semProd.acquire();
        }
        catch (InterruptedException e) {
            System.out.println("InterruptedException caught");
        }
        this.item = item;

        System.out.println("Producer produced item : " + item);
        semCon.release();
    }
}
class Producer implements Runnable {
    Q q;
    Producer(Q q)
    {
        this.q = q;
        new Thread(this, "Producer").start();
    }
    public void run()
    {
        for (int i = 0; i < 5; i++)
            q.put(i);
    }
}
class Consumer implements Runnable {
    Q q;
    Consumer(Q q)
    {
        this.q = q;
        new Thread(this, "Consumer").start();
    }

    public void run()
    {
        for (int i = 0; i < 5; i++)
            q.get();
    }
}
public class SOproductor_consumidor {

    public static void main(String[] args){
        Q q = new Q();

        new Consumer(q);

        new Producer(q);
    }
}
```

2. RESOLUCIÓN “PRODUCTOR – CONSUMIDOR”

2.1. CÓDIGO FUENTE EN JAVA

```java
package com.mycompany.problemapc;

import java.util.LinkedList;
import java.util.Queue;

public class ProblemaPC {

    public static void main(String[] args) {
        Cola colaCompartida = new Cola(5);

        Thread productorThread = new Thread(new Productor(colaCompartida));
        Thread consumidorThread = new Thread(new Consumidor(colaCompartida));

        productorThread.start();
        consumidorThread.start();
    }
}
class Cola {
    private Queue<Integer> cola;
    private int capacidad;

    public Cola(int capacidad) {
        this.cola = new LinkedList<>();
        this.capacidad = capacidad;
    }

    public synchronized void agregar(int elemento) {
        while (cola.size() == capacidad) {
            try {
                wait();
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt();
            }
        }
        cola.add(elemento);
        notifyAll();
    }

    public synchronized int eliminar() {
        while (cola.isEmpty()) {
            try {
                wait();
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt();
            }
        }
        int elemento = cola.remove();
        notifyAll();
        return elemento;
    }
}

class Productor implements Runnable {
    private Cola cola;

    public Productor(Cola cola) {
        this.cola = cola;
    }

    @Override
    public void run() {
```

```
        for (int i = 1; i <= 10; i++) {
            cola.agregar(i);
            System.out.println("Productor agregó: " + i);
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt();
            }
        }
    }
}

class Consumidor implements Runnable {
    private Cola cola;

    public Consumidor(Cola cola) {
        this.cola = cola;
    }

    @Override
    public void run() {
        for (int i = 1; i <= 10; i++) {
            int elemento = cola.eliminar();
            System.out.println("Consumidor eliminó: " + elemento);
            try {
                Thread.sleep(2000);
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt();
            }
        }
    }
}
```

2.2. RESULTADOS

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ ProblemaPC ---
Productor agrego: 1
Consumidor elimino: 1
Productor agrego: 2
Productor agrego: 3
Consumidor elimino: 2
Productor agrego: 4
Productor agrego: 5
Consumidor elimino: 3
Productor agrego: 6
Consumidor elimino: 4
Productor agrego: 7
Productor agrego: 8
Consumidor elimino: 5
Productor agrego: 9
Productor agrego: 10
Consumidor elimino: 6
Consumidor elimino: 7
Consumidor elimino: 8
Consumidor elimino: 9
Consumidor elimino: 10
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  20.838 s
Finished at: 2023-05-12T11:21:43-05:00
------------------------------------------------------------------------
```

3. LINK INFORME – GITHUB (PDF)

https://github.com/mrrows45/so_epis

4. LINK VIDEO - DRIVE

https://drive.google.com/drive/folders/1TiVXt24bu9-rpe15m-O7ccMCZz9XA1je?usp=share_link

5. WEBGRAFÍA

Greyrat, R. (05 de JULIO de 2022). *Solución productor-consumidor utilizando semáforos en Java | conjunto 2*. Obtenido de https://barcelonageeks.com/solucion-productor-consumidor-utilizando-semaforos-en-java-conjunto-2/