Agenda y mensajes aleatorios con elementos de sincronización de hilos

Gustavo García Sánchez

José Rodolfo Reyes Luna

Rodrigo Burciaga

Escuela Superior de Cómputo

Josué Rangel González

3CV8

Contenido

[Introducción 3](#_Toc525837740)

[Objetivo 4](#_Toc525837741)

[Método 4](#_Toc525837742)

[Resultados 5](#_Toc525837743)

[Conclusiones 7](#_Toc525837744)

## Introducción

La aplicación presentada aquí fué realizada con los sistemas de sincronización de Python3.7 los cuales son: Semáforos y Locks.

Un semáforo en Python es un objeto que se maneja con un contador interno el cual se decrementa con cada llamada a la función acquire() e incrementada con cada llamada a release(). El contador no puede nunca estar nunca debajo de cero, cuando acquire() encuentra un cero en el contador, lo bloquea, esperando hasta que alguna otra tarea llame a la función release().

El valor opcional del argumento impone el valor para el contador interno (1 por defecto). Si el valor dado es menor a 0 entonces un ValueError se mostrará.

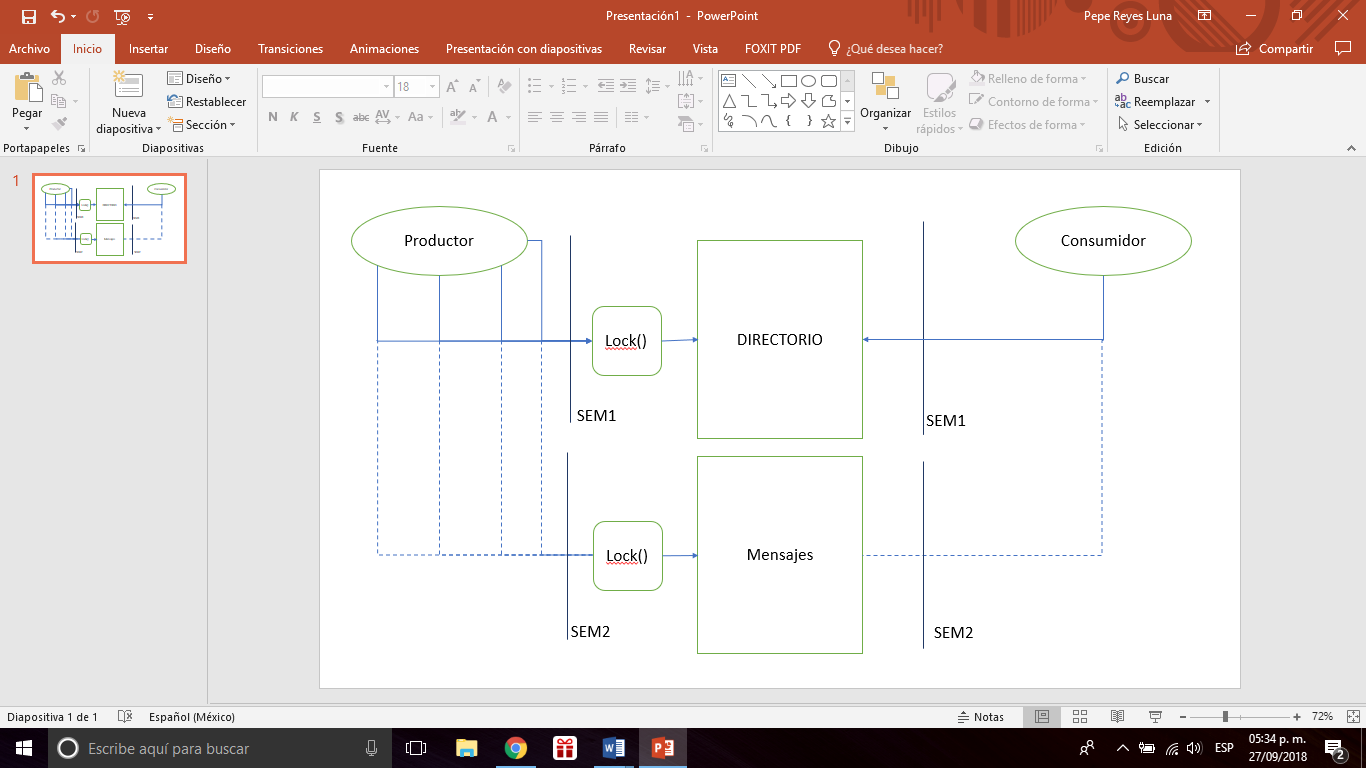
* acquire() : Adquiere el valor del semáforo. Si el valor del contador interno es mayor a cero, decrementa en uno y regresa True inmediatamente. Si es cero, espera a que se haga una llamada a release() y regresa True.
* locked(): Regresa True si el semáforo no puede ser obtenido inmediatamente.
* release(): Libera el semáforo, incrementando el contador interno en uno. Puede despertar una tarea que está esperando a obtener un semáforo.

## Objetivo

El objetivo de la aplicación es mostrar las ventajas de usar hilos y controlar su dependencia de datos a través de la sincronización.

## Método

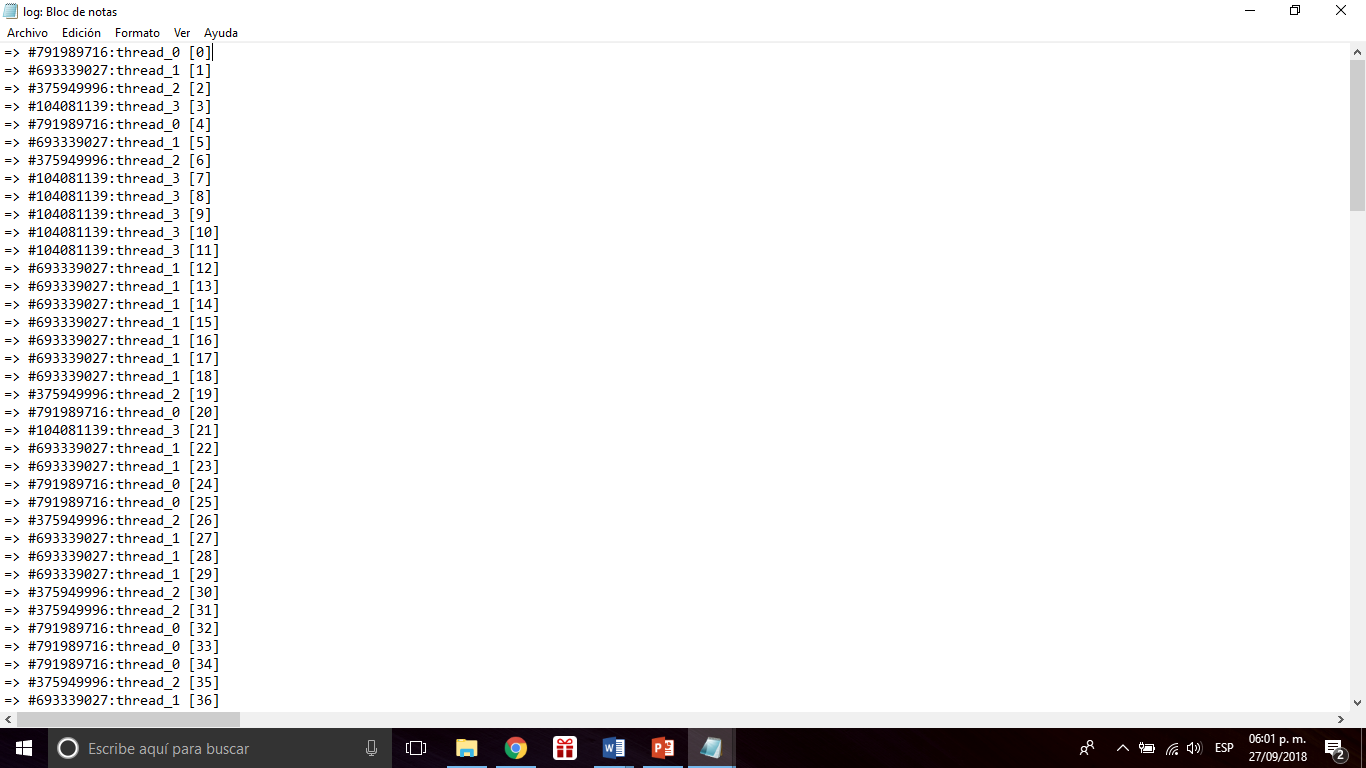
El siguiente diagrama explica el funcionamiento del programa:

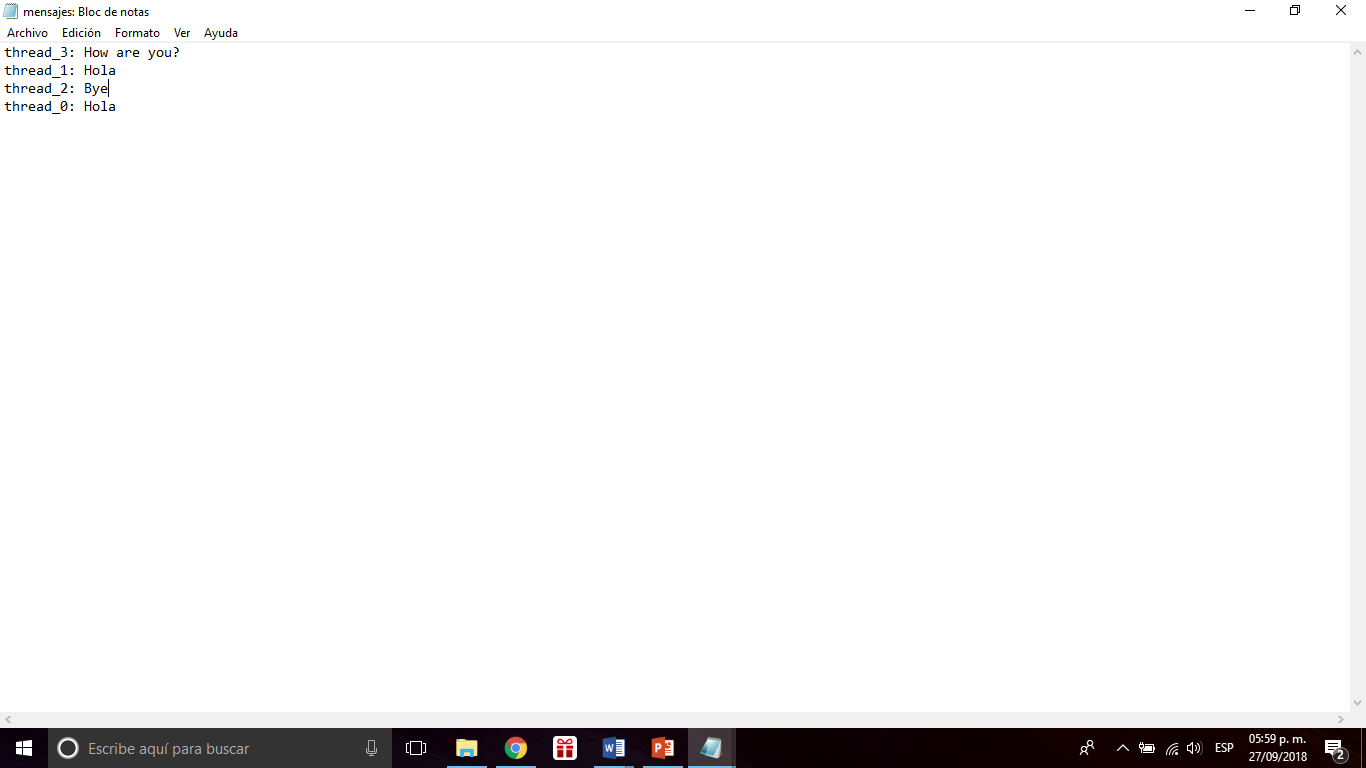


El productor crea 4 hilos, los cuales crearán n registros para el directorio y para los mensajes. Sem1 y Sem2 se inicializan en 0 y el productor les aplicará el método reléase() o suma 1 al contador de los semáforos. El consumidor aplicará en los semáforos el método acquire() que restará 1 al contador de los semáforos; esta acción la ejecutará tantas veces como hilos de productor existan de esta forma el consumidor empezará a leer hasta que todos los productores hayan terminado de escribir. Los métodos lock() en Python equivalen a semáforos binarios, por lo que nos permitirá controlar el flujo de entrada y salida, así de esta forma solo un hilo productor escribirá a la vez.

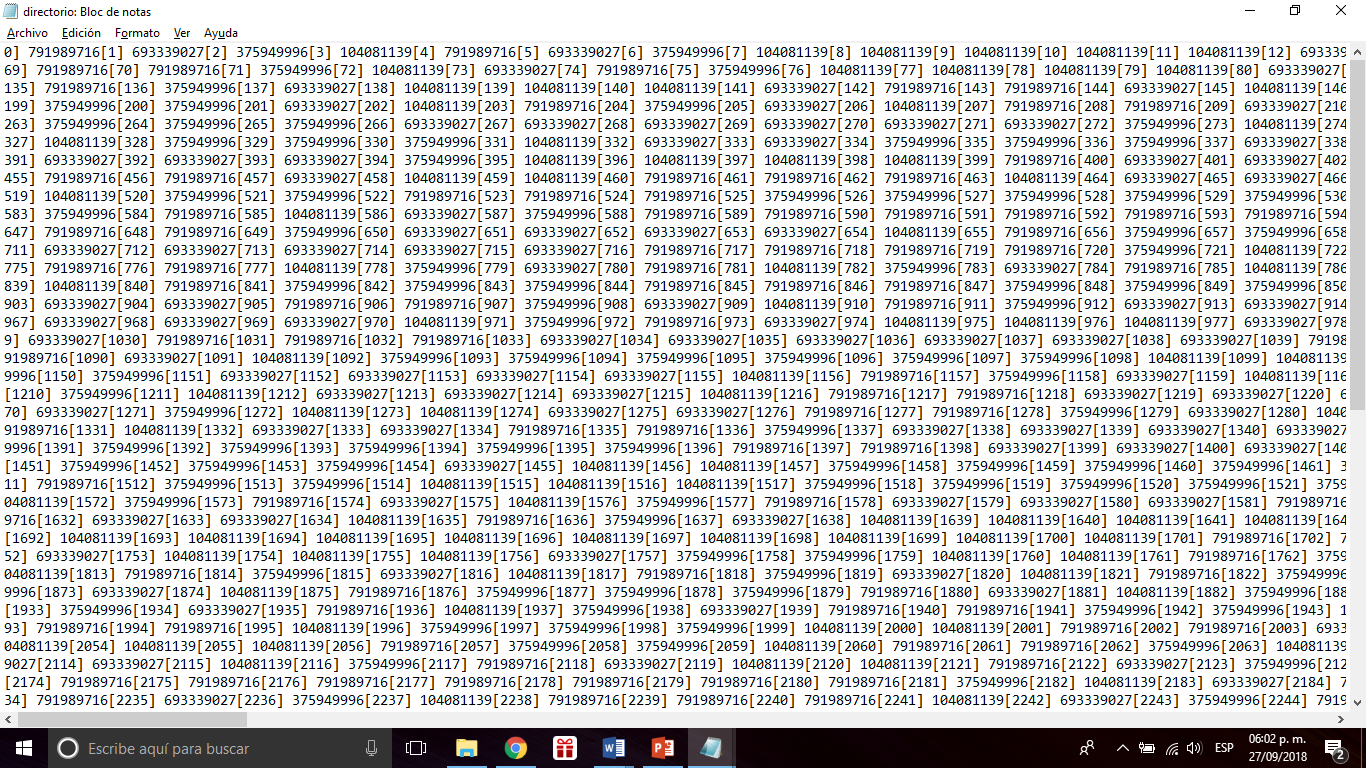
## Resultados

Log



Mensajes

Directorio



## Conclusiones

Reyes Luna José Rodolfo

La implementación de los semáforos y otros mecanismos de sincronización, a diferencia de otros lenguajes de programación, es bastante simple e intuitiva. Los foros están llenos de información por lo que facilitó el realizar la práctica.

García Sánchez Gustavo

El manejo de Hilos y métodos de sincronización es diferente en Python3 a la implementación vista en clase con ejemplos en C. Pero aunque sea diferente la implementación, la lógica de los semáforos para lograr una buena comunicación es la misma.