



**Universidad Nacional Autónoma de
México**

Facultad de Ingeniería



Ingeniería en Computación

Sistemas Operativos

Tarea #1

“Ejercicios de sincronización”

Grupo: 6

Profeso: ING. Gunnar Eyal Wolf Iszaevich

Integrantes:

González Iniestra Emilio

Suarez Guzmán Dayna Yarelly

Fecha de entrega: 22-10-24

Semestre: 2025-1

El problema que decidieron resolver

El de Santa Claus

Planteamiento:

Santa Claus duerme en el Polo Norte mientras sus elfos trabajan frenéticamente en la construcción de millones de nuevos juguetes.

A veces se topan con un problema — Pueden pedir ayuda a Santa Claus, pero sólo de tres en tres.

Sus nueve renos pasan todo el año de vacaciones en las playas del Caribe.

Santa debe despertar a tiempo para iniciar su viaje anual.

Reglas:

Si los nueve renos están de vuelta, es hora de despertar a Santa Claus para que inicie su recorrido.

Si los elfos tienen problemas construyendo algún juguete, le piden ayuda a Santa Claus.

Pero para no darle demasiada lata, lo hacen sólo cuando hay tres elfos que tienen un problema. Mientras tanto, lo dejan dormir.

Puede haber cualquier cantidad de elfos.

El lenguaje y entorno en que lo desarrollaron.

El programa fue desarrollado en Python porque facilita el manejo de hilos con la librería `threading` y ofrece herramientas nativas para la concurrencia. Además, Python es versátil y ya teníamos experiencia utilizándolo, lo que lo convirtió en una buena opción para implementar la sincronización de procesos en este proyecto.

¿Qué tengo que saber / tener / hacer para ejecutar su programa en mi computadora?

Para ejecutar el programa, necesitas tener Python 3 instalado en tu computadora. También debes instalar el paquete `colorama` usando el comando `pip install colorama` para los colores en la salida de consola. Una vez que lo tengas, guarda el código en un archivo `.py` y ejecútalo desde la terminal.

La estrategia de sincronización (mecanismo / patrón) que emplearon para lograrlo

Mecanismo síncrono, utilizamos una combinación de semáforos y barreras para sincronizar los hilos. Los semáforos controlan cuándo Santa Claus se despierta y qué acciones realiza, ya sea ayudar a los elfos o repartir regalos con los renos. Las barreras garantizan que los renos solo actúen cuando todos regresen de vacaciones, y que los elfos pidan ayuda en grupos de tres. Además, un `mutex` asegura que Santa realice una acción a la vez, evitando condiciones de carrera.

Si están implementando alguno de los refinamientos

Sí, refinamos el programa usando un mutex para que Santa no pueda realizar varias tareas simultáneamente. Esto mejora la sincronización al asegurar que solo se enfoque en una tarea a la vez. También optimizamos el uso de barreras para coordinar a los renos y los elfos de manera eficiente, evitando posibles problemas de concurrencia.

Cualquier duda que tengan. Si notan que algún pedazo de la implementación podría mejorar, si algo no les terminó de quedar bien...

Al momento de hacer el código nos costo trabajo que sea de forma síncrona ya que al momento de imprimir los elfos son muchos y parece que Santa no descansa y los renos van llegando pero no respeta que son 9 renos y tiene que salir a repartir regalos

```
In [1]: runfile('C:/Users/daysg/Downloads/Tarea1_GonzalezEm...')
Santa: 🛷 Durmiendo...

Elfo 14: ¡Tengo un problema con los juguetes! 🧝
Elfo 16: ¡Tengo un problema con los juguetes! 🧝
Elfo 17: ¡Tengo un problema con los juguetes! 🧝
Elfo 23: ¡Tengo un problema con los juguetes! 🧝

Santa: 🛖 Desperté!

Elfo 10: ¡Tengo un problema con los juguetes! 🧝
Elfo 24: ¡Tengo un problema con los juguetes! 🧝
Elfo 28: ¡Tengo un problema con los juguetes! 🧝
Elfo 26: ¡Tengo un problema con los juguetes! 🧝

Santa: ¡Tres elfos tienen problemas, voy a ayudar! 🧝🧝🧝

Elfo 16: ¡Gracias Santa por la ayuda! 🧝
Elfo 17: ¡Gracias Santa por la ayuda! 🧝
Elfo 14: ¡Gracias Santa por la ayuda! 🧝

Elfo 1: ¡Tengo un problema con los juguetes! 🧝

Console 52/A X
Reno 6: ¡He vuelto de vacaciones! 🦌
Reno 8: ¡He vuelto de vacaciones! 🦌
Reno 9: ¡He vuelto de vacaciones! 🦌
Elfo 5: ¡Tengo un problema con los juguetes! 🧝
Elfo 13: ¡Tengo un problema con los juguetes! 🧝
Elfo 18: ¡Tengo un problema con los juguetes! 🧝
Elfo 7: ¡Tengo un problema con los juguetes! 🧝
Elfo 8: ¡Tengo un problema con los juguetes! 🧝
Elfo 30: ¡Tengo un problema con los juguetes! 🧝
Santa: 🛷 Durmiendo...

Santa: 🛖 Desperté!

Reno 1: ¡He vuelto de vacaciones! 🦌
Reno 5: ¡He vuelto de vacaciones! 🦌
Elfo 2: ¡Tengo un problema con los juguetes! 🧝
Elfo 3: ¡Tengo un problema con los juguetes! 🧝
Elfo 11: ¡Tengo un problema con los juguetes! 🧝
Elfo 15: ¡Tengo un problema con los juguetes! 🧝
Elfo 6: ¡Tengo un problema con los juguetes! 🧝
```

```

Elfo 16: ¡Tengo un problema con los juguetes! 🧸
Santa: ¡Tres elfos tienen problemas, voy a ayudar! 🧑🏻🦋🧸
Elfo 3: ¡Gracias Santa por la ayuda! 🧸
Elfo 11: ¡Gracias Santa por la ayuda! 🧸
Elfo 29: ¡Gracias Santa por la ayuda! 🧸

Reno 2: ¡He vuelto de vacaciones! 🦌
Elfo 2: ¡Tengo un problema con los juguetes! 🧸
Elfo 7: ¡Tengo un problema con los juguetes! 🧸

Elfo 30: ¡Tengo un problema con los juguetes! 🧸
Elfo 17: ¡Tengo un problema con los juguetes! 🧸

Santa: 🦌 Durmiendo...
Santa: 🦌 Desperté!

Elfo 29: ¡Tengo un problema con los juguetes! 🧸
Santa: ¡Todos los renos están de vuelta, es hora de repartir regalos! 🦌🎁
Santa: ¡Reparti todos los regalos, ahora a dormir! 😴

```

Codigo:

```
"""
```

Tarea 1

Gonzalez Iniestra Emilio

Suarez Guzman Dayna Yarelly

```
"""
```

```
import threading
```

```
import time
```

```
import random
```

```
import colorama
```

```
# Parámetros
```

```
num_renos = 9
```

```
num_elfos = 30
```

```
elfos_por_grupo = 3
```

```
# Semáforos y mutex para control
```

```
santa_sem = threading.Semaphore(0)
```

```
renos_sem = threading.Semaphore(0)
```

```
elfos_ayuda_sem = threading.Semaphore(0)
```

```
renos_listos = threading.Semaphore(0)
```

```
elfos_listos = threading.Semaphore(0)
```

```

# Barreras para sincronización
renosBarrera = threading.Barrier(num_renos)
elfosBarrera = threading.Barrier(elfos_por_grupo)

# Mutex para acceso seguro a recursos compartidos
mutex = threading.Semaphore(1)

class SantaClaus:
    def __init__(self):
        self.mi_color = colorama.Fore.RED + colorama.Back.WHITE

    def imprimir(self, mensaje):
        print(self.mi_color + "Santa: " + mensaje + "\n" + colorama.Fore.RESET +
              colorama.Back.RESET)

    def dormir(self):
        while True:
            self.imprimir("💤 Durmiendo...")
            santa_sem.acquire() # Espera a que los renos o elfos lo despierten
            self.imprimir("👴 Desperté!")
            time.sleep(1) # Retardo para simular que Santa toma su tiempo para despertarse
            self.accion_santa() # Realiza la acción que corresponda

    def accion_santa(self):
        # Santa debe ayudar a los elfos o repartir regalos sin secuencias
        mutex.acquire()

        # Despertar basado en cuál semáforo se libera primero
        if renos_listos.acquire(blocking=False):
            self.repartir_regalos() # Reparte regalos cuando los 9 renos estén listos
        elif elfos_listos.acquire(blocking=False):
            self.ayudar_elfos() # Ayuda a los 3 elfos cuando están listos

        mutex.release()

    def repartir_regalos(self):
        self.imprimir("¡Todos los renos están de vuelta, es hora de repartir regalos! 🦌📺")
        for _ in range(num_renos):
            renos_sem.release() # Libera a los renos para repartir regalos

```

```

for _ in range(num_renos):
    renos_sem.acquire()

self.imprimir("¡Repartí todos los regalos, ahora a dormir! 🤪")
time.sleep(2) # Retardo después de repartir regalos

def ayudar_elfos(self):
    self.imprimir("¡Tres elfos tienen problemas, voy a ayudar! 🧝 🌲")
    for _ in range(elfos_por_grupo):
        elfos_ayuda_sem.release() # Libera a los elfos para que Santa los ayude
    time.sleep(2) # Retardo después de ayudar a los elfos

class Reno:
    def __init__(self, id_reno):
        self.id = id_reno
        self.mi_color = colorama.Fore.GREEN + colorama.Back.WHITE

    def imprimir(self, mensaje):
        print(self.mi_color + "Reno {}: {} \n".format(self.id + 1, mensaje) +
              colorama.Fore.RESET + colorama.Back.RESET)

    def vacaciones(self):
        time.sleep(random.randint(5, 10)) # Simula las vacaciones de manera aleatoria
        self.imprimir("¡He vuelto de vacaciones! 🌴")

# Todos los renos deben llegar a la barrera
renosBarrera.wait() # Espera a que todos los renos lleguen
renos_listos.release() # Despierta a Santa si todos los renos están de vuelta
santa_sem.release() # Despierta a Santa

renos_sem.acquire() # Espera a que Santa los libere para repartir regalos
self.imprimir("¡Repartiendo regalos con Santa! 🎁")
renos_sem.release() # Notifica a Santa que ha terminado de repartir
time.sleep(1) # Retardo para que el proceso no ocurra de inmediato

class Elfo:
    def __init__(self, id_elfo):

```

```

self.id = id_elfo
self.mi_color = colorama.Fore.BLUE + colorama.Back.WHITE

def imprimir(self, mensaje):
    print(self.mi_color + "Elfo {}: {}".format(self.id + 1, mensaje) + "\n" +
colorama.Fore.RESET + colorama.Back.RESET)

def trabajar(self):
    while True:
        time.sleep(random.randint(2, 6)) # Simula el trabajo de los elfos
        self.imprimir("¡Tengo un problema con los juguetes! 🧸 ")

        # Todos los elfos deben llegar a la barrera en grupos de 3
        elfosBarrera.wait() # Espera a que tres elfos necesiten ayuda
        elfos_listos.release() # Despierta a Santa cuando tres elfos tienen problemas
        santa_sem.release() # Despierta a Santa

        elfos_ayuda_sem.acquire() # Espera a que Santa lo ayude
        self.imprimir("¡Gracias Santa por la ayuda! 🎄 ")
        time.sleep(1) # Retardo para simular que los elfos siguen trabajando después

def iniciar_renos():
    for i in range(num_renos):
        reno = Reno(i)
        threading.Thread(target=reno.vacaciones).start()

def iniciar_elfos():
    for i in range(num_elfos):
        elfo = Elfo(i)
        threading.Thread(target=elfo.trabajar).start()

def main():
    colorama.init(autoreset=True)

    # Iniciar Santa Claus
    santa = SantaClaus()
    threading.Thread(target=santa.dormir).start()

    # Iniciar renos y elfos

```

```
iniciar_renos()
iniciar_elfos()
```

```
# El programa sigue corriendo indefinidamente simulando el trabajo en el Polo Norte
while True:
    time.sleep(1)
```

```
if __name__ == "__main__":
    main()
```