LABORATORIO 1 – GESTIÓN DE PROCESOS

Axel Amarilla
Ingeniería en Sistemas
Sistemas Operativos
2025

Objetivo del laboratorio

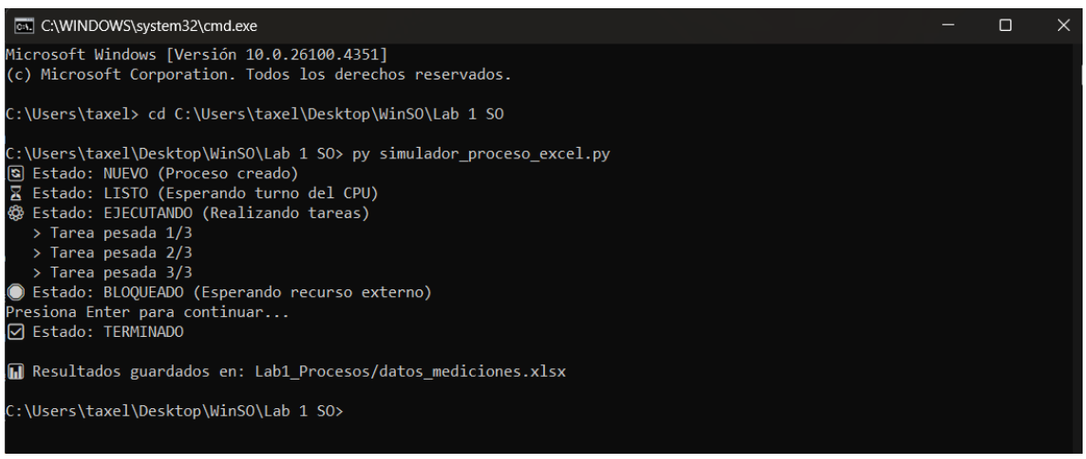
Simular y analizar el comportamiento de los procesos en un sistema operativo, identificando sus diferentes estados, tiempos de transición y comportamiento frente a la planificación del CPU.

Simulación de estados de procesos

Se creó un programa en Python que representa los estados fundamentales del ciclo de vida de un proceso:

- Nuevo: creación del proceso
- Listo: espera para ser ejecutado
- Ejecutando: realiza operaciones
- Bloqueado: espera por un recurso externo
- Terminado: finaliza su ejecución

Durante la ejecución del programa, se observaron estos estados de forma secuencial.



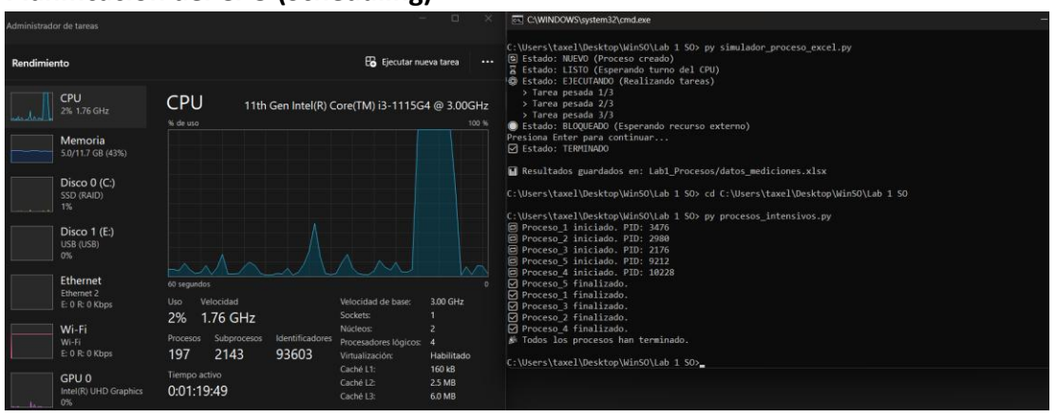
Aquí se muestra la salida del programa indicando claramente los estados del proceso en tiempo real, simulando pausas y bloqueos como lo haría el sistema operativo.

Medición de tiempos

El programa registra automáticamente los tiempos entre cada transición y los guarda en un archivo Excel (datos_mediciones.xlsx). Esto permite visualizar el comportamiento temporal del proceso.

Transición	Tiempo (segundos)
NUEVO -> LISTO	1.001
LISTO -> EJECUTANDO	1.001
EJECUTANDO -> BLOQUEADO	3.003
BLOQUEADO -> TERMINADO	47.726
DURACIÓN TOTAL	53.732

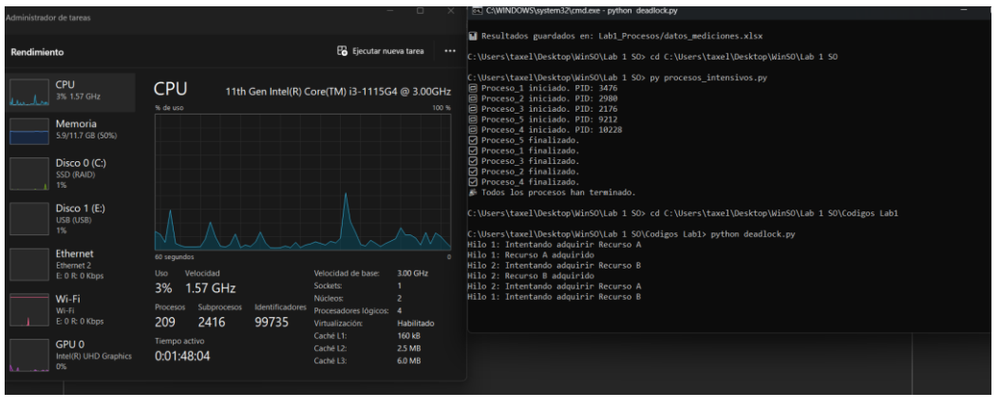
Planificación del CPU (Scheduling)



Se pusieron en marcha 5 tareas a la vez que hicieron muchos cálculos al mismo tiempo. Mientras se ejecutaban, noté que el procesador (CPU) se usaba mucho, lo que significa que el sistema de la computadora repartió bien el tiempo entre cada tarea, tal como estaba programado para hacerlo.

Simulación de Deadlock

Se ejecutó un script en Python (deadlock.py) que simula una situación de interbloqueo (deadlock) mediante el uso de dos hilos que intentan adquirir recursos en distinto orden. El objetivo fue observar el comportamiento del sistema operativo cuando se presenta este tipo de conflicto en la asignación de recursos.



Durante la ejecución del script deadlock.py, se observó que el uso de CPU, memoria y disco se mantuvo estable. El sistema operativo no presentó bloqueos ni fallos visibles. El interbloqueo quedó confinado a los hilos del programa, sin afectar a otros procesos del sistema.

Intentar Resolver el deadlock:

```
C:\Users\taxel\Desktop\WinSO\Lab 1 SO> cd C:\Users\taxel\Desktop\WinSO\Lab 1 SO\Codigos Lab1

C:\Users\taxel\Desktop\WinSO\Lab 1 SO\Codigos Lab1> python deadlock.py
Hilo 1: Intentando adquirir Recurso A
Hilo 1: Recurso A adquirido
Hilo 2: Intentando adquirir Recurso B
Hilo 2: Recurso B adquirido
Hilo 2: Intentando adquirir Recurso A
Hilo 1: Intentando adquirir Recurso B
```

Como se puede observar, cuando se activa un deadlock, dos o más procesos quedan esperando indefinidamente recursos que están siendo usados por el otro. Cada uno bloquea parcialmente al otro, y ninguno puede continuar, lo que provoca una detención total del programa en esa parte.

Conclusión

Este laboratorio ayudó a entender cómo funcionan los procesos dentro del sistema operativo. Al hacer las pruebas, se pudo ver cómo se comportan los programas, cómo se reparten los recursos y qué pasa cuando ocurre un bloqueo. Fue útil para ver en la práctica lo que se estudia en teoría.