



Facultad de Ingeniería

Carrera: Ingeniería en Sistemas

Asignatura:  
Sistemas Operativos

Catedrático:  
Ing. Elmer Danuary Padilla Sorto .

Ángeles Izamar Euceda Herrera 20221930061

Comayagua, Comayagua, 22 de abril de 2025

## **Introducción**

La planificación de procesos es una función fundamental de los sistemas operativos modernos. Este proyecto tiene como finalidad simular distintos algoritmos de planificación, analizar su comportamiento y medir su eficiencia mediante métricas clave. La simulación permite comprender cómo se gestionan los recursos de CPU y cuál es el impacto de cada algoritmo sobre los procesos en ejecución.

## **Objetivo del Proyecto**

- Desarrollar un simulador interactivo en C++ con interfaz gráfica (Windows Forms) que implemente los algoritmos de planificación de procesos FCFS, SJF y Round Robin, y que permita observar visualmente el orden de ejecución, el uso de CPU y las métricas asociadas a cada proceso.

## **Algoritmos Implementados**

- FCFS (First Come First Serve): Atiende los procesos en el orden en que llegan.
- SJF (Shortest Job First): Selecciona el proceso con el menor tiempo de rafaga.
- Round Robin: Asigna un tiempo fijo (quantum) a cada proceso y rota entre ellos hasta finalizar.

Cada algoritmo se puede seleccionar desde la interfaz gráfica y visualizar su comportamiento con los datos ingresados por el usuario.

## **Detalles de Implementación**

→ Lenguaje utilizado: C++

→ Entorno de desarrollo: Visual Studio 2022

→ Interfaz: Windows Forms

→ Entrada de procesos: Manual, mediante campos para ID, tiempo de llegada, rafaga y prioridad.

→ Salida: Diagrama de Gantt textual, métricas individuales y promedio general.

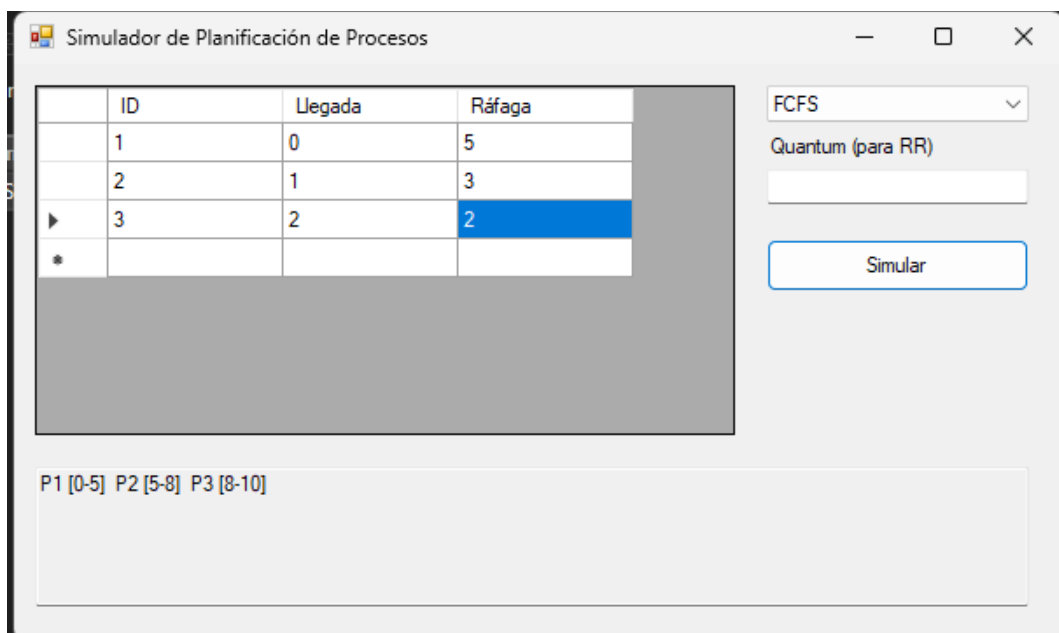
## **Métricas Calculadas**

- Tiempo de espera: Tiempo en que un proceso espera en cola antes de ser atendido.
- Tiempo de respuesta: Tiempo desde la llegada hasta la primera vez que se atiende el proceso.
- Tiempo de retorno: Tiempo total desde la llegada hasta que finaliza el proceso.
- Uso de CPU: Proporción del tiempo en que la CPU estuvo activa.

## Casos de Prueba

Se ingresaron distintos conjuntos de procesos para validar el comportamiento de los algoritmos. En todos los casos, se observaron los tiempos correctamente y se generaron los diagramas esperados. Se evaluó también el funcionamiento del quantum en Round Robin y su impacto en la alternancia de procesos.

### Caso 1



Atiende a los procesos en el orden de llegada, sin interrupciones. El primer proceso que llega se ejecuta hasta terminar luego el siguiente, y así sucesivamente.

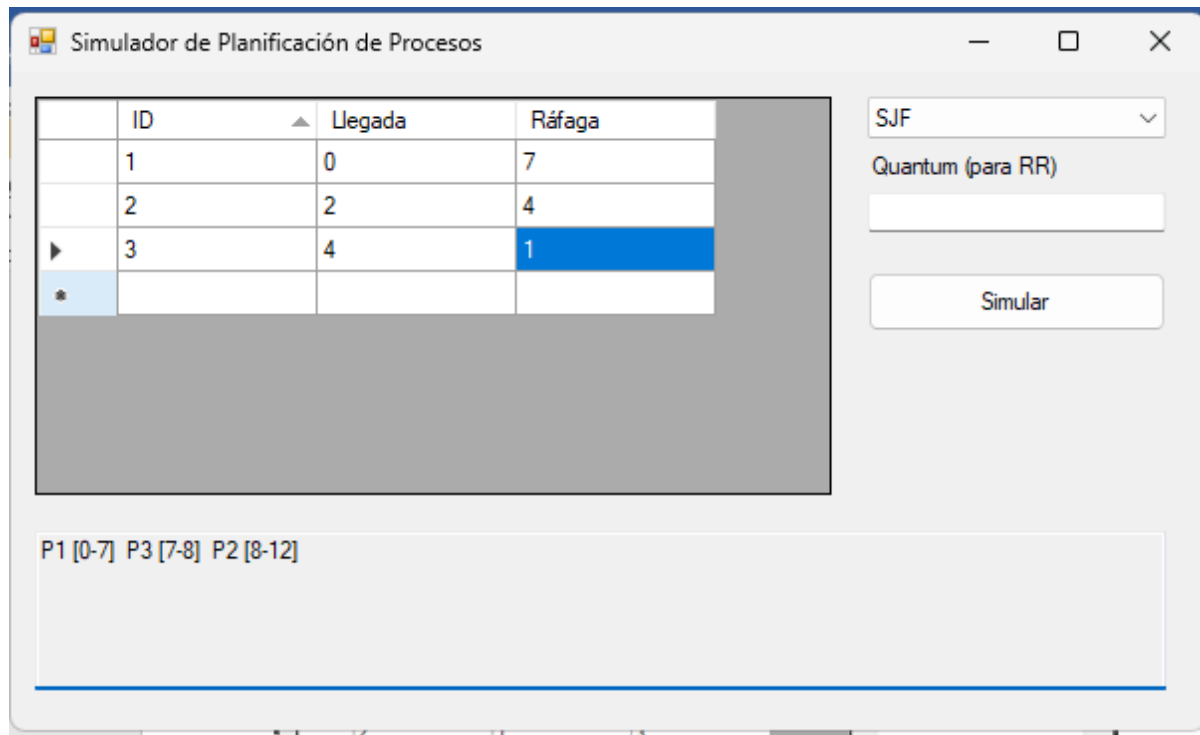
→ Simulación paso a paso:

P1 llega en 0 y se ejecuta durante 5 → [0-5]

P2 ya ha llegado (llegó en 1) → [5-8]

P3 también llegó antes, se ejecuta después → [8-10]

## Caso 2



Elige el proceso con la ráfaga más corta entre los que ya han llegado. No interrumpe procesos (en su versión no-preemptiva).

→ Simulación paso a paso:

Solo P1 está disponible al inicio, así que empieza.

P1 se ejecuta → [0-7]

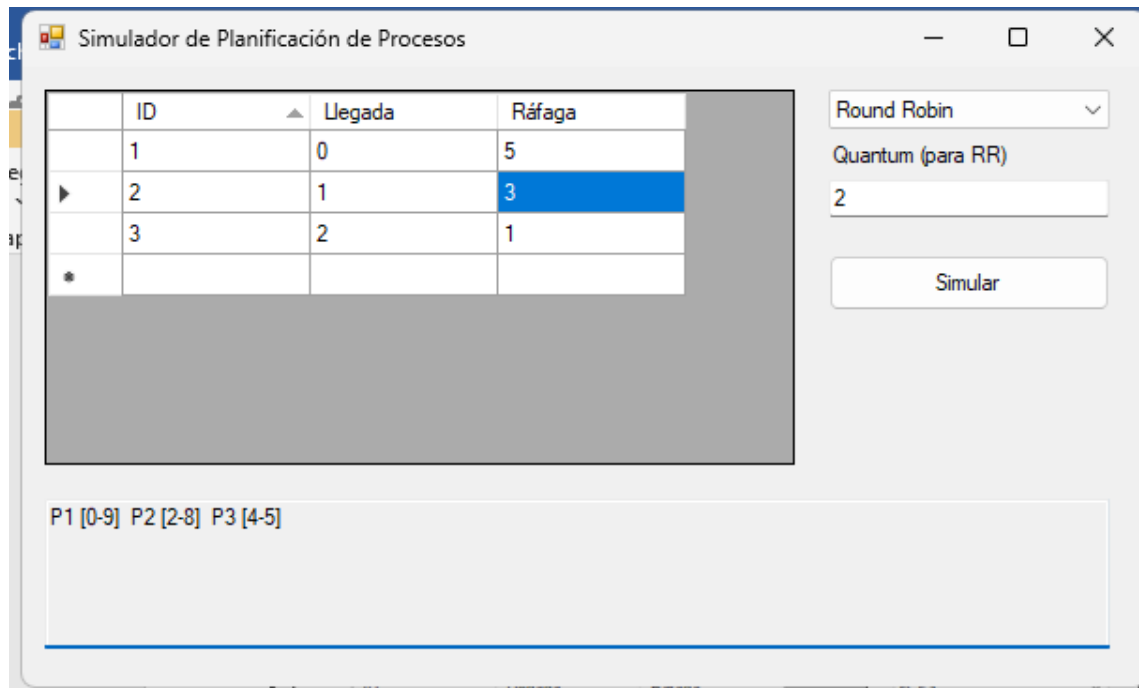
En ese tiempo, llegan P2 y P3.

De ellos, P3 tiene la ráfaga más corta (1) → [7-8]

Queda P2, con ráfaga 4 → [8-12]



### Caso 3



	ID	Llegada	Ráfaga
	1	0	5
▶	2	1	3
	3	2	1
*			

Round Robin

Quantum (para RR)

2

Simular

P1 [0-9] P2 [2-8] P3 [4-5]

Asigna a cada proceso un quantum fijo (aquí: 2). Si el proceso no termina en ese tiempo, se interrumpe y vuelve a la cola luego elige el próximo proceso disponible por orden de llegada. Cabe recalcar que esta versión no muestra el Gantt en detalle, sino que muestra intervalos acumulados: p.inicio: el momento en que el proceso se ejecutó por primera vez. Y p.fin: el momento en que terminó su última ejecución.

→ Simulación paso a paso:

- P1 (llega en 0): ejecuta 2 unidades → queda 3
- P2 (llega en 1): ejecuta 2 unidades → queda 1
- P3 (llega en 2): ejecuta 1 unidad → termina
- P1 (vuelve): ejecuta 2 unidades → queda 1
- P2 (vuelve): ejecuta 1 unidad → termina
- P1 (última vuelta): ejecuta 1 unidad → termina

## **Conclusiones**

Este simulador permite comprender de manera visual y práctica el impacto de distintos algoritmos de planificación sobre el rendimiento del sistema. La interfaz amigable facilita la introducción de datos y el análisis de resultados, mientras que la implementación modular permite la futura extensión hacia algoritmos como Prioridades o MLFQ. El proyecto refuerza los conceptos teóricos vistos en clase mediante la programación aplicada.

## **Anexo**

<https://youtu.be/ry2QDoGjqCY> link de presentación de proyecto