
Universidad Nacional Autónoma de México

Facultad de Ingeniería


# TAREA 02. COMPARACION DE PLANIFICADORES

Alumno: Martínez García José Eduardo (320304679)

Profesor: Dr. Gunnar Eyal Wolf Iszaevich

Sistemas Operativos

Grupo teoría: 08

Semestre 2026-1

Fecha de entrega: 28/11/25

## Objetivos

**Tarea “Planificadores”**

escribir un programa que genere *varias cargas aleatorias*, y compare el resultado sobre varias ejecuciones. Presentar cinco ejecuciones, para poder comparar las tendencias, revisen manualmente por lo menos algunos de los resultados, para confirmar que son correctos.

## Contenido

Se resolvieron algoritmos de planificación de procesos. Estos algoritmos permiten asignar tiempos de espera, proporciones de penalización, y otras métricas fundamentales para evaluar el rendimiento de la planificación.

El programa debe:

- Generar múltiples procesos con tiempos de llegada y duraciones aleatorias.
- Simular los algoritmos de planificación: FCFS, Round Robin (RR) y Shortest Process Next (SPN).
- Evaluar las métricas clave de cada algoritmo.
- Mostrar visualizaciones que indiquen cómo cada algoritmo asigna el tiempo de CPU a los procesos.

Lenguaje: python3

### Algoritmos Realizados

- FCFS (First Come, First Served):

Se implementó un algoritmo propio que recibe como parámetros el número de rondas y un conjunto de procesos. Planifica los procesos en el orden en que llegan, siendo su principal ventaja la simplicidad, aunque puede presentar inanición si existen procesos con tiempos largos.

- RR1 y RR4 (Round Robin con quantum variable):

Este algoritmo recibe también un conjunto de procesos, un número de rondas y un parámetro q que representa el quantum o intervalo de tiempo asignado a cada proceso.

En el caso de RR1 se usa un quantum = 1 y en RR4 se usa un quantum = 4.
La principal ventaja de este método es la equidad en la asignación del CPU, ya que todos los procesos reciben atención en ciclos regulares.

- SPN (Shortest Process Next):

Prioriza los procesos con menor tiempo de ejecución.
Su ventaja es que minimiza el tiempo promedio de espera, aunque puede provocar inanición en procesos largos.

## Métricas Calculadas

Tiempo de T:

Tiempo total que un proceso toma desde su llegada hasta su finalización.

$$Fórmula: T = tiempo\_final - tiempo\_llegada.$$

## Tiempo de Espera (E):

Tiempo que un proceso pasa esperando para ser ejecutado. o

$$Fórmula: E = T - t.$$

## Proporción de Penalización (P):

o Relación entre el tiempo de y la duración del $proceso$.

$$Fórmula: P = T / t.$$

**Ejecución del Programa**

El programa está desarrollado en Python, bajo un paradigma estructurado, por lo que no requiere compilación. Para ejecutarlo basta con abrir la terminal y ejecutar el archivo principal:

```
$ python3 MartínezJosé_Tarea2.py
```

**Estructura del Código**

## 1. Clases y funciones principales:

- Proceso: clase que modela un proceso (nombre, llegada, duración, duración restante).
- generarProcesos(): crea los procesos con tiempos aleatorios.
- fcfs(), rr(), spn(): simulan los algoritmos de planificación.
- metricas(): calcula T, E y P.
- esquemaVisual(): genera una representación visual del orden de ejecución.

## 2. Flujo del programa:

1. Generar procesos.
2. Ejecutar cada algoritmo.
3. Calcular métricas.
4. Mostrar resultados y esquema visual.
5. Repetir por 5 rondas.

## 3. Parámetros configurables:

- Número de procesos.
- Tiempo máximo de llegada y duración.
- Quantum para Round Robin.
-

## Pruebas de escritorio

RONDA 1

```
--- RONDA 1 ---
A: llegada=0, duracion=2
B: llegada=1, duracion=3
C: llegada=5, duracion=2
D: llegada=7, duracion=1
E: llegada=12, duracion=5

Esquema visual de FCFS:

|A|A|B|B|B|C|C|D|E|E|E|E|E|


FCFS -> T=2.80, E=0.20, P=3.51

Esquema visual de RR (Quantum=1):

|A|A|B|B|B|C|C|D|E|E|E|E|E|


RR (Quantum=1) -> T=2.80, E=0.20, P=3.51

Esquema visual de RR (Quantum=4):

|A|A|B|B|B|C|C|D|E|E|E|E|E|


RR (Quantum=4) -> T=2.80, E=0.20, P=3.51

Esquema visual de SPN:

|A|A|B|B|B|C|C|D|E|E|E|E|E|

SPN -> T=2.80, E=0.20, P=3.51
```

RONDA 2

```
--- RONDA 2 ---
A: llegada=0, duracion=2
B: llegada=1, duracion=3
C: llegada=5, duracion=1
D: llegada=12, duracion=6
E: llegada=17, duracion=6

Esquema visual de FCFS:

|A|A|B|B|B|C|D|D|D|D|D|D|E|E|E|E|E|E|


FCFS -> T=4.00, E=0.40, P=3.13

Esquema visual de RR (Quantum=1):

|A|A|B|B|B|C|D|D|D|D|D|D|E|E|E|E|E|E|


RR (Quantum=1) -> T=4.00, E=0.40, P=3.13

Esquema visual de RR (Quantum=4):

|A|A|B|B|B|C|D|D|D|D|D|D|E|E|E|E|E|E|


RR (Quantum=4) -> T=4.00, E=0.40, P=3.13

Esquema visual de SPN:

|A|A|B|B|B|C|D|D|D|D|D|D|E|E|E|E|E|E|


SPN -> T=4.00, E=0.40, P=3.13
```

RONDA 3

```
--- RONDA 3 ---
A: llegada=0, duracion=4
B: llegada=7, duracion=5
C: llegada=13, duracion=6
D: llegada=18, duracion=6
E: llegada=21, duracion=3

Esquema visual de FCFS:

|A|A|A|A|B|B|B|B|B|C|C|C|C|C|C|D|D|D|D|D|D|E|E|E|


FCFS -> T=5.80, E=1.00, P=4.01

Esquema visual de RR (Quantum=1):

|A|A|A|A|B|B|B|B|B|C|C|C|C|C|C|D|D|D|E|D|E|D|E|D|


RR (Quantum=1) -> T=6.20, E=1.40, P=4.05

Esquema visual de RR (Quantum=4):

|A|A|A|A|B|B|B|B|B|C|C|C|C|C|C|D|D|D|D|D|D|E|E|E|


RR (Quantum=4) -> T=5.80, E=1.00, P=4.01

Esquema visual de SPN:

|A|A|A|A|B|B|B|B|B|C|C|C|C|C|C|D|D|D|D|D|D|E|E|E|


SPN -> T=5.80, E=1.00, P=4.01
```

RONDA 4

```
--- RONDA 4 ---
A: llegada=0, duracion=7
B: llegada=3, duracion=1
C: llegada=8, duracion=1
D: llegada=11, duracion=5
E: llegada=18, duracion=6

Esquema visual de FCFS:

|A|A|A|A|A|A|A|B|C|D|D|D|D|D|E|E|E|E|E|E|


FCFS -> T=4.80, E=0.80, P=5.04

Esquema visual de RR (Quantum=1):

|A|A|A|A|B|A|A|A|C|D|D|D|D|D|E|E|E|E|E|E|


RR (Quantum=1) -> T=4.40, E=0.40, P=4.47

Esquema visual de RR (Quantum=4):

|A|A|A|A|A|A|A|B|C|D|D|D|D|D|E|E|E|E|E|E|


RR (Quantum=4) -> T=4.80, E=0.80, P=5.04

Esquema visual de SPN:

|A|A|A|A|A|A|A|B|C|D|D|D|D|D|E|E|E|E|E|E|


SPN -> T=4.80, E=0.80, P=5.04
```

RONDA 5

```
--- RONDA 5 ---
A: llegada=0, duracion=2
B: llegada=7, duracion=6
C: llegada=12, duracion=7
D: llegada=19, duracion=2
E: llegada=23, duracion=1

Esquema visual de FCFS:

|A|A|B|B|B|B|B|B|C|C|C|C|C|C|C|D|D|E|


FCFS -> T=4.00, E=0.40, P=8.20

Esquema visual de RR (Quantum=1):

|A|A|B|B|B|B|B|B|C|C|C|C|C|C|C|D|D|E|


RR (Quantum=1) -> T=4.00, E=0.40, P=8.20

Esquema visual de RR (Quantum=4):

|A|A|B|B|B|B|B|B|C|C|C|C|C|C|C|D|D|E|


RR (Quantum=4) -> T=4.00, E=0.40, P=8.20

Esquema visual de SPN:

|A|A|B|B|B|B|B|B|C|C|C|C|C|C|C|D|D|E|


SPN -> T=4.00, E=0.40, P=8.20
```

## conclusión

Después de desarrollar y ejecutar el programa de simulación en Python, pude comprender de manera práctica cómo los diferentes algoritmos de planificación de procesos influyen en el rendimiento de un sistema operativo.

El algoritmo FCFS resultó ser el más simple en su implementación, aunque evidenció tiempos de espera elevados para ciertos procesos. El método Round Robin, en sus variantes RR1 y RR4, mostró una distribución más justa del tiempo de CPU, pero también quedó claro que su eficiencia depende directamente del valor del quantum seleccionado. Por otro lado, el algoritmo SPN ofreció mejores tiempos promedio de ejecución, aunque puede generar inanición en procesos con mayor duración.

Al comparar sus resultados y visualizar las métricas obtenidas, comprendí cómo pequeños ajustes en la planificación pueden modificar significativamente el comportamiento general del sistema, creo que es además un conjunto de implementaciones que te ayudan a comprender mejor como es que funciona internamente nuestro procesador, la razón del movimiento del mouse a pesar de que un sistema en tu computadora se ralentice.