**Informe – Simulador de Planificación de Procesos (SIGET)**

**Manuela García Soto**

**Objetivos**

El propósito de este trabajo es **diseñar e implementar un simulador de planificación de procesos** que permita observar cómo diferentes algoritmos (FIFO, SJF, Round-Robin y Prioridad) gestionan un conjunto de tareas del sistema SIGET.  
El objetivo principal es analizar qué algoritmo ofrece un mejor equilibrio entre:

* **Tiempo de respuesta rápido** para eventos críticos (emergencias de movilidad).
* **Eficiencia global** en la administración de datos masivos.

De esta manera, se busca aplicar los conceptos de planificación vistos en la plataforma y comprender su impacto en un escenario práctico.

**Decisiones de diseño**

* **Lenguaje:** Python, por su facilidad de prototipado y legibilidad.
* **Modelo de proceso:** Se creó la clase Proceso con atributos: llegada, ráfaga, prioridad y tamaño de datos.
* **Estados representados:** Nuevo, Listo, En ejecución, Bloqueado y Terminado.
* **Algoritmos implementados:**
  + **FIFO (First In, First Out):** ejecución según orden de llegada.
  + **SJF (Shortest Job First, no expropiativo):** selecciona primero el proceso con menor ráfaga.
  + **Round-Robin (RR):** planificación con reparto equitativo de tiempo (quantum=2).
  + **Prioridad (expropiativo):** selecciona siempre el proceso con mayor prioridad (número más bajo).
* **Salidas generadas:** trazas en consola mostrando la evolución de cada proceso y métricas promedio: turnaround, tiempo de respuesta y espera.

**Algoritmos y observaciones**

1. **FIFO**
   * Ventaja: muy simple de implementar.
   * Desventaja: procesos cortos esperan demasiado si llegan después de uno muy largo.
   * Observación: en SIGET, puede ser ineficiente para alertas de emergencia.
2. **SJF**
   * Ventaja: minimiza el tiempo de espera promedio.
   * Desventaja: si procesos largos siguen llegando, pueden quedar relegados (inanición).
   * Observación: adecuado para tareas de análisis rutinario en SIGET, pero riesgoso para emergencias críticas.
3. **Round-Robin**
   * Ventaja: reparte equitativamente el procesador, mejora la sensación de respuesta.
   * Desventaja: mayor overhead por cambios de contexto, turnaround más alto en procesos largos.
   * Observación: útil para manejar múltiples solicitudes de usuarios o tareas concurrentes en SIGET.
4. **Prioridad (expropiativo)**
   * Ventaja: asegura que emergencias se atiendan de inmediato.
   * Desventaja: riesgo de inanición en procesos de baja prioridad.
   * Observación: en SIGET, este algoritmo es clave para dar prioridad al tráfico en tiempo real o incidentes críticos.

**Conclusiones**

* **No existe un algoritmo único que sea óptimo en todos los escenarios.**
* **FIFO** es demasiado rígido para un sistema como SIGET, ya que no distingue entre tareas críticas y rutinarias.
* **SJF** optimiza la eficiencia, pero no garantiza respuesta rápida a emergencias.
* **Round-Robin** es más justo y útil para mantener fluidez en múltiples procesos, aunque sacrifica algo de eficiencia.
* **Prioridad** es el más adecuado para SIGET, dado que asegura atención inmediata a eventos críticos de movilidad. Sin embargo, es recomendable complementarlo con técnicas como **aging** para evitar que procesos de baja prioridad nunca se ejecuten.

En conclusión, la mejor estrategia para SIGET sería **combinar algoritmos**: usar **Prioridad** para emergencias y **Round-Robin** para el procesamiento rutinario de datos, logrando así un balance entre eficiencia global y capacidad de reacción ante incidentes.