**UNIVERSIDAD MAYOR DE SAN ANDRÉS**

FACULTAD DE CIENCIAS PURAS Y NATURALES

**CARRERA DE INFORMÁTICA**



**ALGORITMOS DE ADMINISTRACIÓN DEL PROCESADOR Y POLÍTICAS DE GESTIÓN DE MEMORIA**

**Universitario:  CI:**

-Charca Condori Ronaldo  10070217 L.P.

**Materia:** Sistemas Operativos

**Docente:** Lic. Rubén Alcón López

**Paralelo: ” B ”**

**La Paz -Bolivia**

**2024**

**ÍNDICE**

1. Introducción ...................................................................................................................... 3

2. Descripción del Conjunto de Algoritmos y Políticas Simulados ...................................... 4

2.1. Administración del Procesado ..................................................................................... 4

2.2. Políticas de Gestión de Memoria ............................................................................. 4-5

2.3. Configuración de la Simulación .............................................................................. 5-6

2.4. Generación de Entradas ............................................................................................... 6

3. Descripción de la Entrada Generada .............................................................................. 6-7

4. Descripción y diseño de la estrategia ................................................................................ 7

4.1. FIFO (First In, First Out) ............................................................................................ 8

4.2. SJN (Shortest Job Next) .............................................................................................. 9

4.3. Prioridad (Priority Scheduling) .............................................................................. 9-10

4.4. Asignación Contigua Simple ..................................................................................... 11

4.5. Partición Fija (Primer Ajuste, Mejor Ajuste, Peor Ajuste) .................................. 12-13

4.6. Particionada Reubicable ............................................................................................ 13

5. Ejecución de los procesos ............................................................................................... 14

5.1. Administración del Procesador ................................................................................. 14

5.2. Políticas de Gestión de Memoria ......................................................................... 15-17

6. Conclusiones y Recomendaciones .................................................................................. 17

6.1. Conclusión de Gestión de Procesos .......................................................................... 17

6.2. Conclusión de Gestión de Memoria .................................................................... 17-18

6.3. Recomendaciones ..................................................................................................... 18   
7. Referencias Bibliográficas……………...……………………………………………… 18

**1. Introducción.**

En los sistemas operativos modernos, la gestión eficiente de los recursos, como el procesador y la memoria, es esencial para garantizar un rendimiento óptimo y una experiencia de usuario satisfactoria. Para lograr este objetivo, se emplean diversos algoritmos de planificación de procesos y políticas de asignación de memoria, cada uno diseñado para manejar diferentes patrones de carga de trabajo y requisitos del sistema.

Este proyecto tiene como objetivo simular un conjunto de estos algoritmos y políticas, evaluando su desempeño en términos de eficiencia, utilización de recursos y manejo de fragmentación. En particular, se considerarán los algoritmos de planificación de procesos FIFO, SJN y Prioridad, así como las políticas de gestión de memoria: asignación contigua simple, asignación particionada fija (con estrategias de Primer Ajuste, Mejor Ajuste y Peor Ajuste) y asignación particionada reubicable.

Esta simulación tiene como objetivo principal comprender cómo estos algoritmos y políticas afectan el rendimiento de los sistemas operativos, particularmente en términos de planificación de procesos y asignación de memoria.

**2. Descripción del Conjunto de Algoritmos y Políticas Simulados.**

**2.1. Administración del Procesador.**

Los algoritmos que se simularán son los siguientes:

-FIFO (First In, First Out):

Este algoritmo procesa las tareas en el orden en que llegan, sin considerar su duración. Es simple pero puede ser ineficiente en presencia de tareas largas.

-SJN (Shortest Job Next):

Prioriza las tareas que requieren menor tiempo de ejecución. Este enfoque minimiza el tiempo promedio de espera pero requiere conocer de antemano el tiempo de cada tarea.

-Prioridad (Priority Scheduling):

Cada tarea se procesa según su prioridad asignada. Puede ser con prioridad estática o dinámica. Si dos tareas tienen la misma prioridad, se utiliza otro criterio (como FIFO) para desempatar.

**2.2. Políticas de Gestión de Memoria.**

Las políticas de asignación de memoria que se simularán son las siguientes:

-Asignación Contigua Simple:

Todo el espacio de memoria disponible se asigna a un único proceso. Es sencilla, pero no permite ejecutar múltiples procesos simultáneamente.

-Asignación Particionada Fija:

La memoria se divide en particiones de tamaño fijo. Se implementarán los siguientes enfoques para asignar procesos a estas particiones:

Primer Ajuste: Asigna el proceso a la primera partición disponible que sea lo suficientemente grande.

-Mejor Ajuste: Busca la partición más pequeña que sea lo suficientemente grande para acomodar el proceso, minimizando el desperdicio de memoria.

-Peor Ajuste: Asigna el proceso a la partición más grande disponible, dejando grandes fragmentos libres.

-Asignación Particionada Reubicable:

Permite mover los procesos en memoria para compactar los espacios libres y acomodar nuevos procesos. Esto reduce la fragmentación externa, aunque introduce un costo adicional de reubicación.

La simulación de estos algoritmos y políticas permitirá analizar aspectos clave como:

-Rendimiento: Tiempo promedio de espera, tiempo de respuesta y utilización del procesador.

-Eficiencia: Uso óptimo de la memoria disponible y manejo de la fragmentación.

-Adaptabilidad: Respuesta a diferentes patrones de carga de trabajo.

Este análisis servirá para identificar cuál combinación de algoritmos y políticas es más adecuada bajo diferentes condiciones y requisitos del sistema operativo.

**2.3. Configuración de la Simulación.**

--Unificación de Recursos:

Las peticiones de procesos se generarán de forma que sean compatibles tanto con los algoritmos de planificación del procesador como con las políticas de gestión de memoria. Cada proceso contendrá información suficiente para ser manejado por ambos grupos de algoritmos.

-Datos por Proceso:

Cada proceso incluirá los siguientes atributos:

-Hora de llegada: Momento en el que el proceso entra al sistema.

-Tiempo de ejecución: Duración que el proceso requiere para completarse (para algoritmos de planificación del procesador).

-Prioridad: Nivel de prioridad asignado al proceso (para algoritmos basados en prioridad).

Luego tenemos otro algoritmo que genera:

Tamaño del proceso: Cantidad de memoria que el proceso requiere (para políticas de gestión de memoria).

Tamaño de memoria: Cantidad de la memoria que sirve para almacenar los procesos.

El tiempo de ejecución será generado con valores en un rango predefinido para representar procesos cortos y largos.

La prioridad será un valor entero en un rango definido (por ejemplo, de 1 a 10), donde un valor menor indica mayor prioridad.

El tamaño del proceso será un valor aleatorio dentro de los límites de la memoria disponible, dependiendo del tipo de simulación (asignación simple, particionada fija o reubicable).

**2.4. Generación de Entradas.**

Todos estos datos serán utilizados en los algoritmos, ya que al generar una cierta cantidad de datos estos servirán para los algoritmos como el FIFO, SJN, PRIORIDAD, Asignación continua simple, Asignación particionada(simple, mejor, peor), Asignación Particionada Reubicable.

Configuración de Memoria:

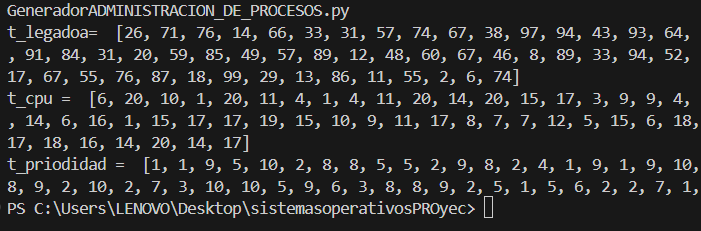
La memoria será definida en términos de su capacidad total y el tipo de partición utilizada. Por ejemplo:

Asignación contigua simple: Una única partición que cubre toda la memoria.

Asignación particionada fija: Memoria dividida en particiones de tamaño fijo.

Asignación particionada reubicable: Se permiten movimientos para compactar espacios libres.

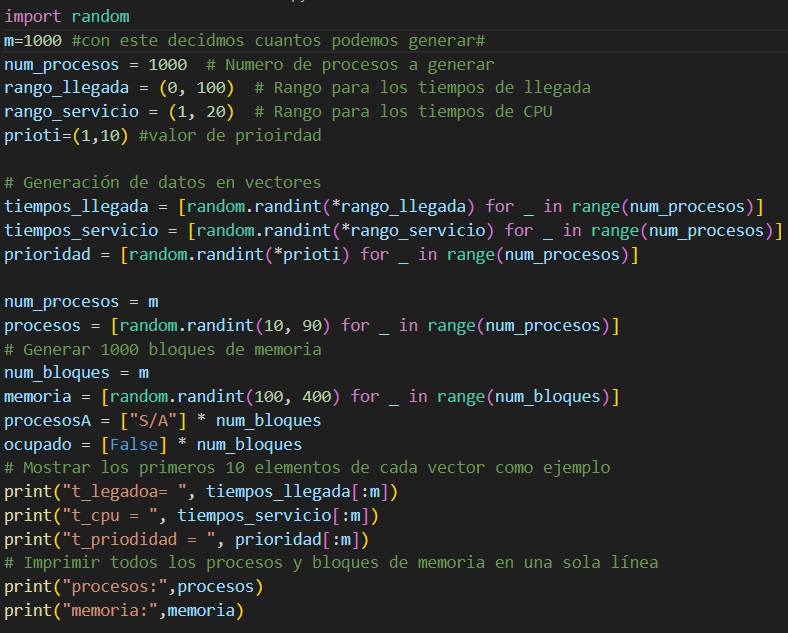
**3. Descripción de la Entrada Generada.**





**4. Descripción y diseño de la estrategia.**

Se tiene una función que analiza todos estos datos lo algoritmos



**4.1.FIFO (First In, First Out).**

Este algoritmo se basa en el principio de cola: el primer proceso que llega es el primero en ejecutarse. No toma en cuenta otros factores como el tiempo de ejecución o la prioridad.

Función:

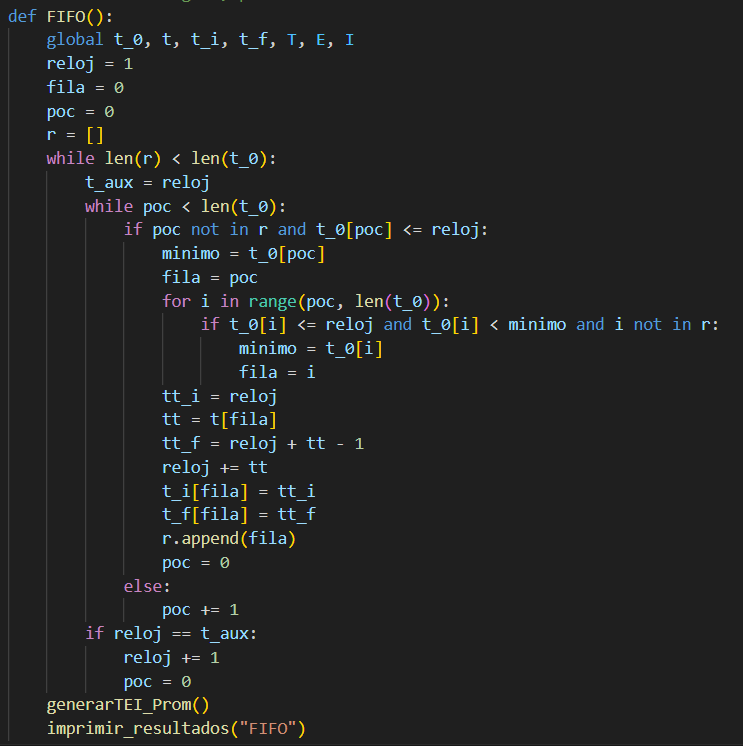
Procesa los trabajos en el orden en que llegan. Es simple, pero puede generar un problema conocido como convoy effect, donde los procesos pequeños esperan mucho si un proceso largo llega primero.

Uso en la simulación:

Ordena los procesos por tiempo de llegada.

Calcula el tiempo inicial y final de cada proceso según el orden de llegada.

-Código del Algoritmo FIFO:



**4.2.SJN (Shortest Job Next).**

Este algoritmo selecciona para su ejecución el proceso con el menor tiempo de ejecución que ya haya llegado al sistema. Es óptimo en cuanto a tiempo promedio de espera, pero puede causar injusticia hacia procesos largos.

Función:

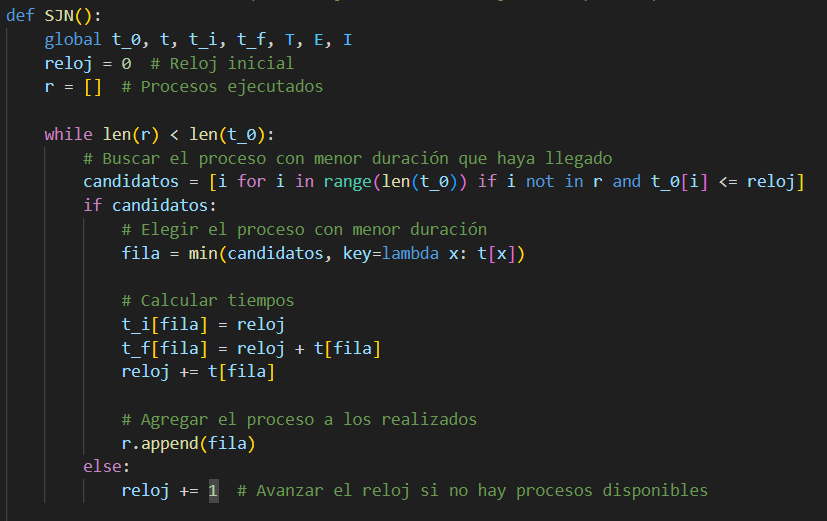
Prioriza los procesos más cortos que ya están listos para ejecutarse, minimizando el tiempo de espera global.

Uso en la simulación:

Ordena dinámicamente los procesos disponibles por su tiempo de ejecución.

Elige el trabajo más corto en cada unidad de tiempo simulada.

-Código del algoritmo SJN:



**4.3. Prioridad (Priority Scheduling).**

Este algoritmo asigna a cada proceso una prioridad numérica, y los procesos con mayor prioridad se ejecutan primero.

Función:

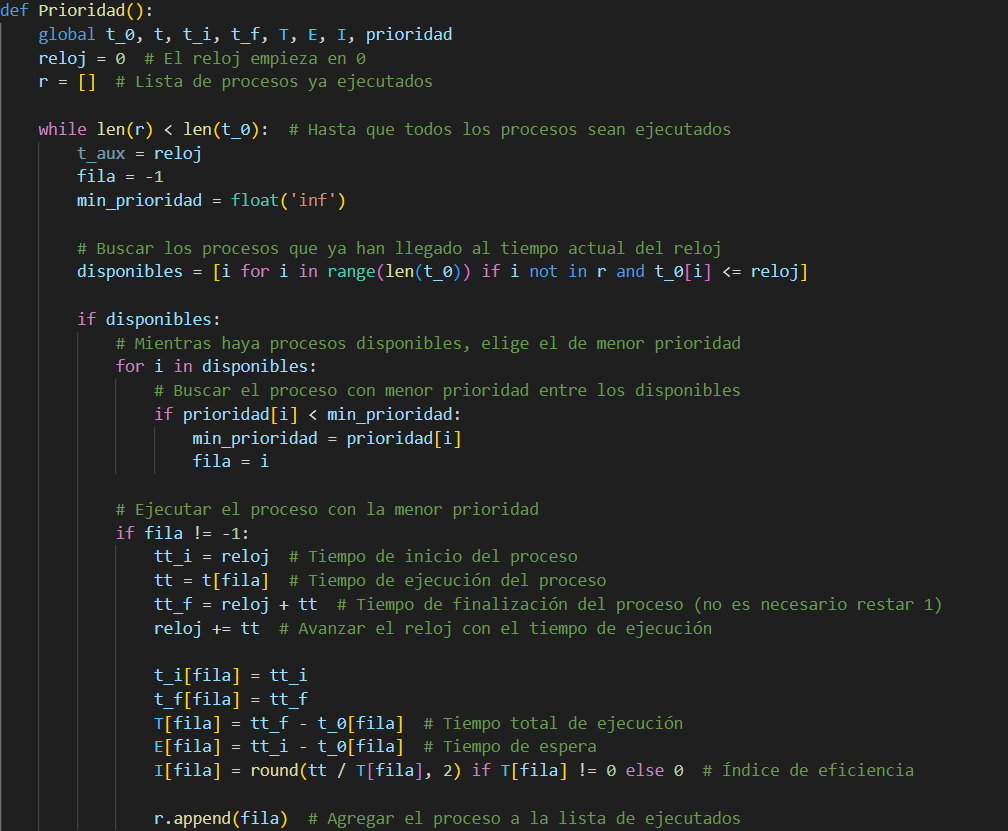
Ordena los procesos según su nivel de prioridad. Si dos procesos tienen la misma prioridad, utiliza un criterio secundario, como el tiempo de llegada.

Uso en la simulación:

Asigna y evalúa prioridades a cada proceso.

Selecciona el proceso con la mayor prioridad en cada ciclo.

-Código de la algoritmo Prioridad:



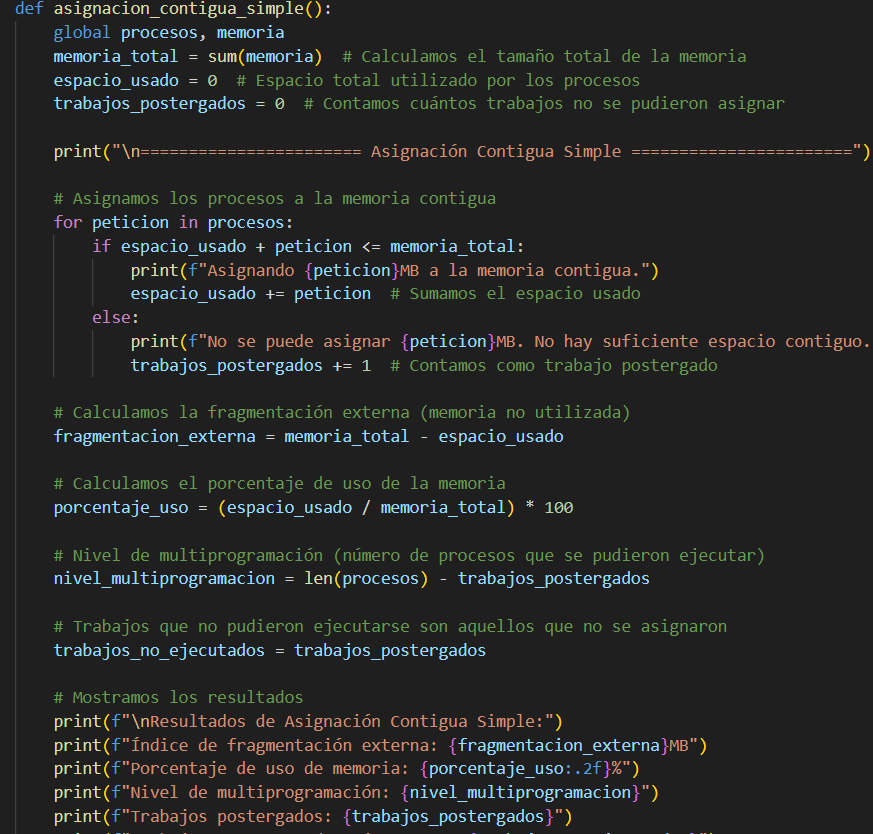
**4.4. Asignación Contigua Simple.**

Asigna los procesos en bloques contiguos dentro de la memoria. Si no hay suficiente espacio contiguo para una solicitud, esta se pospone.

Función: Asigna memoria de manera secuencial, revisando si hay suficiente espacio continuo para cada solicitud.

Uso en la simulación: Permite evaluar la eficiencia de la asignación contigua, destacando la fragmentación externa y el porcentaje de memoria utilizada.

-Código del Algoritmo de Asignación continua simple:



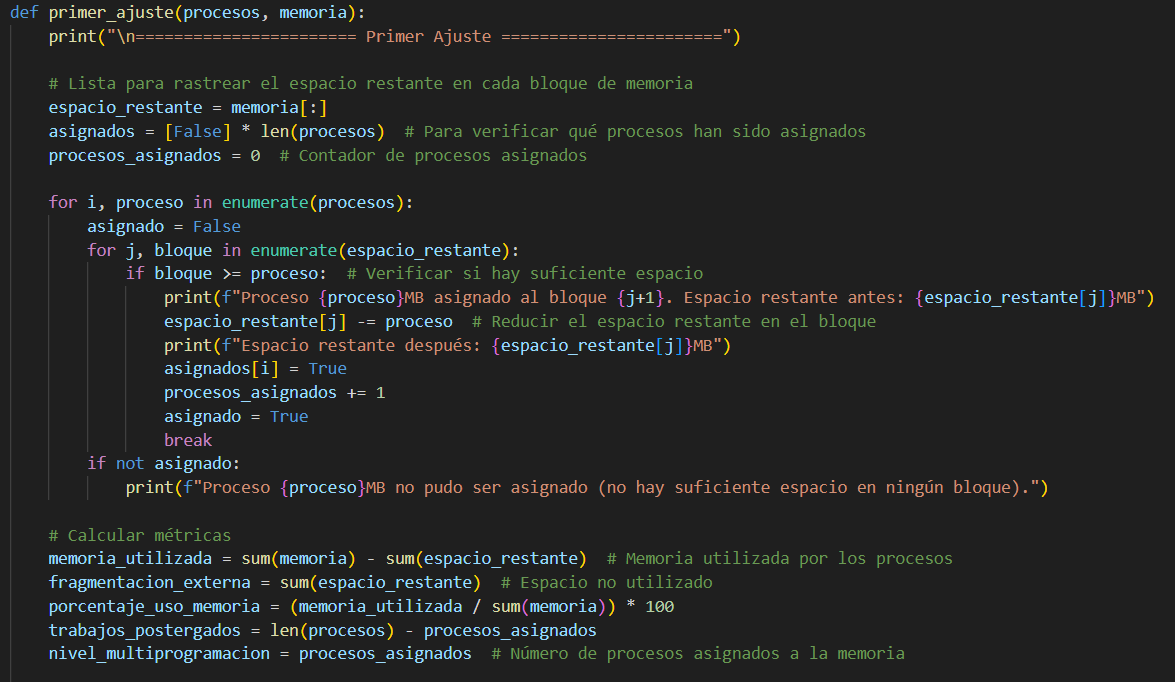
**4.5. Partición Fija (Primer Ajuste, Mejor Ajuste, Peor Ajuste).**

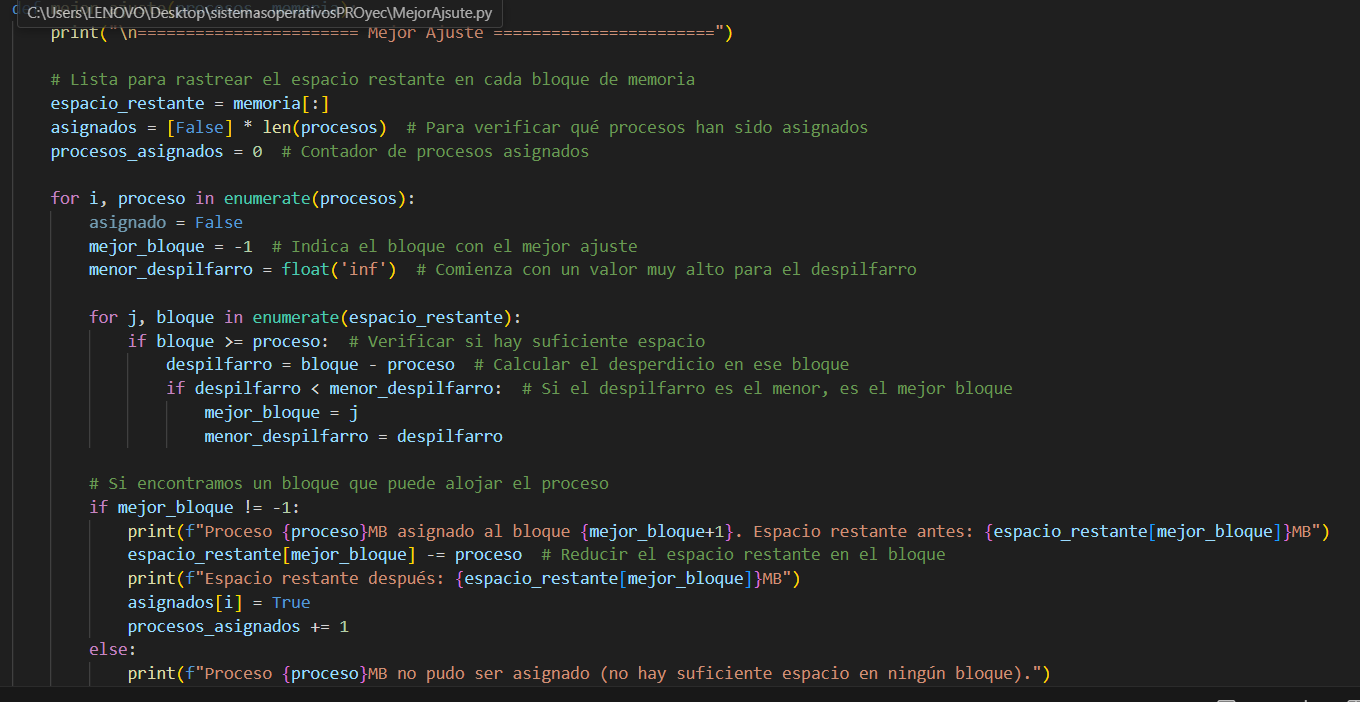
Descripción: Divide la memoria en bloques fijos. Los procesos se asignan a los bloques disponibles utilizando tres algoritmos: Primer Ajuste (el primero que encaje), Mejor Ajuste (el más pequeño que encaje) y Peor Ajuste (el más grande que encaje).

Función: Gestiona la memoria dividiéndola en bloques estáticos y asignando procesos según distintos criterios de ajuste.

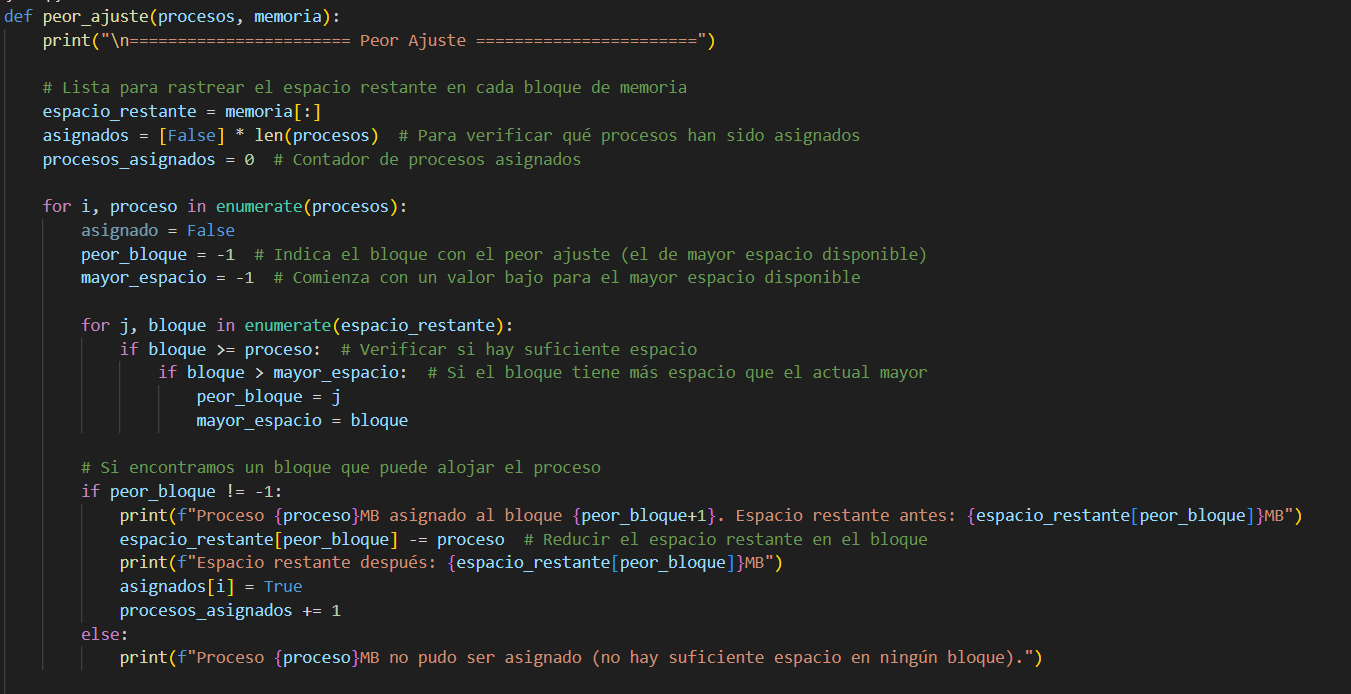
Uso en la simulación: Permite comparar cómo cada algoritmo maneja la asignación de memoria, la fragmentación y el uso de espacio en la memoria.

-Código del algoritmo de Primer ajuste:



-Código del algoritmo de Mejor Ajuste:

-Código del algoritmo de Peor Ajuste:

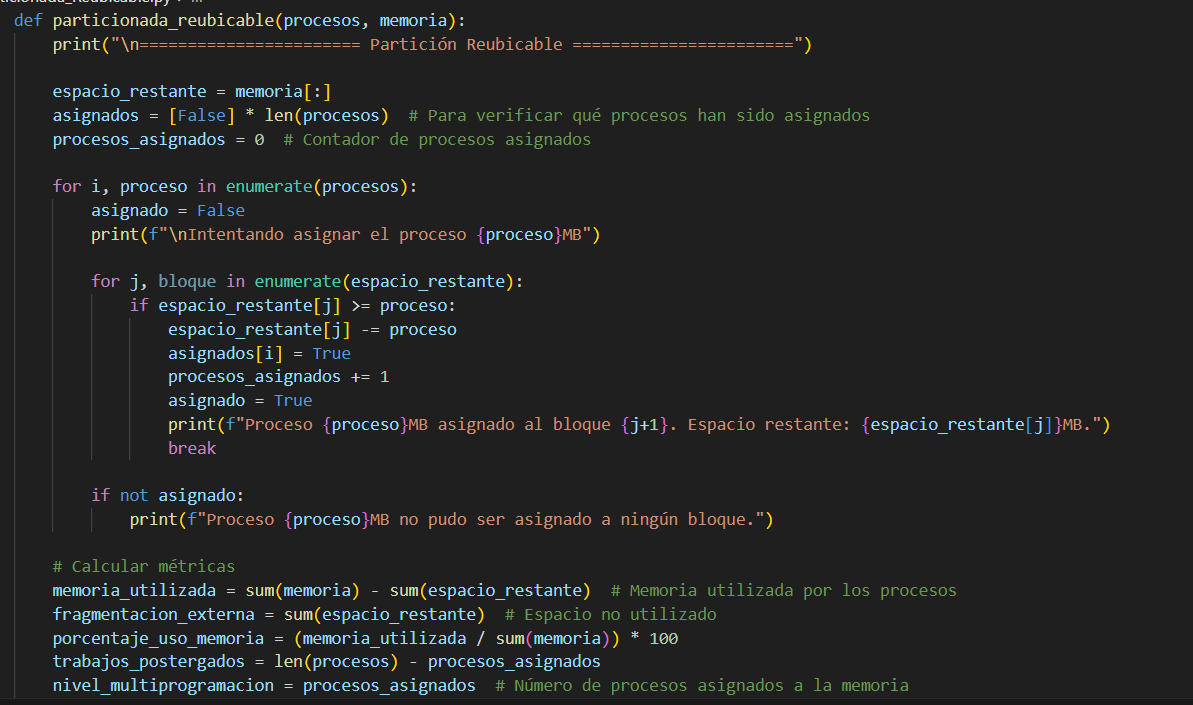


**4.6. Particionada Reubicable.**

Descripción: La memoria se gestiona de manera dinámica, permitiendo la reubicación de procesos en bloques libres para optimizar el uso del espacio.

Función: Asigna procesos a cualquier bloque libre disponible, permitiendo la reubicación para reducir la fragmentación.

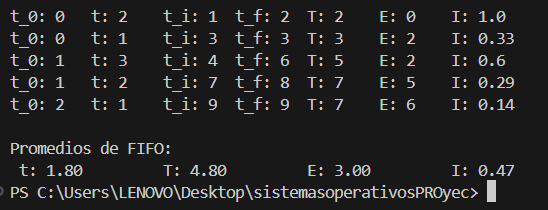
Uso en la simulación: Evalúa la eficiencia de la reubicación de procesos y el manejo de la memoria en situaciones de alta fragmentación.

-Código del Algoritmo de Partición Reubicable:

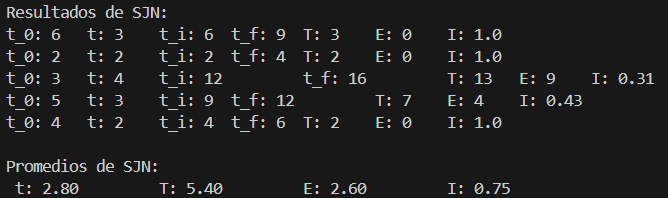
**5.Ejecucion de los procesos.**

**5.1. Administración del Procesador.**

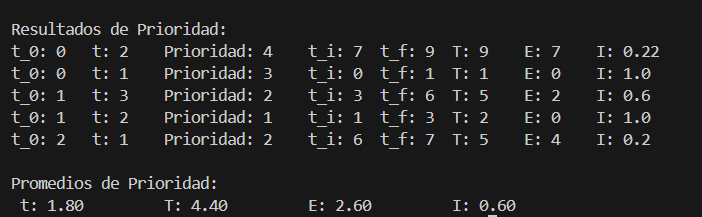
-FIFO:



-JSN:

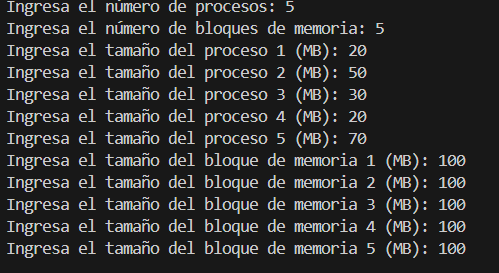


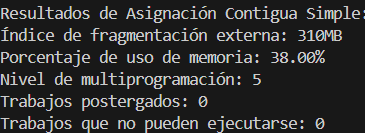
-PRIORIDAD:



**5.2. Políticas de gestión de memoria.**

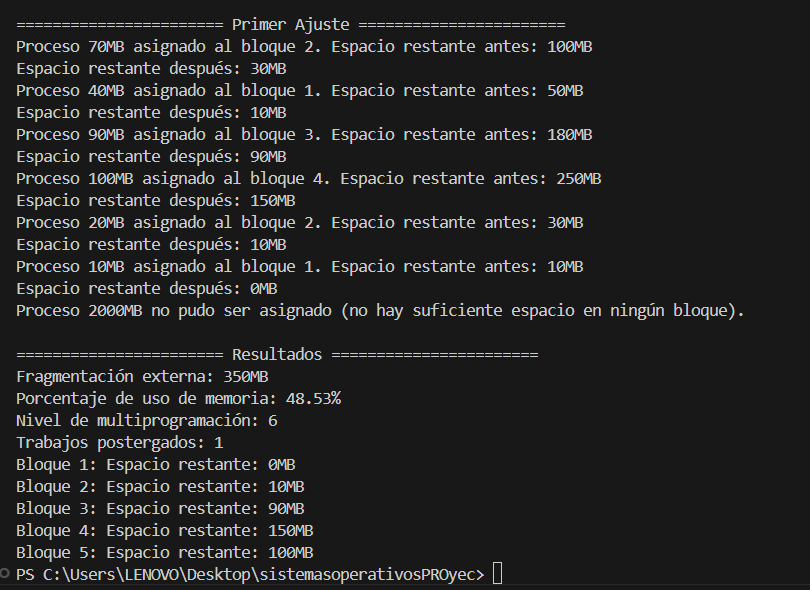
-Asignación continua Simple:



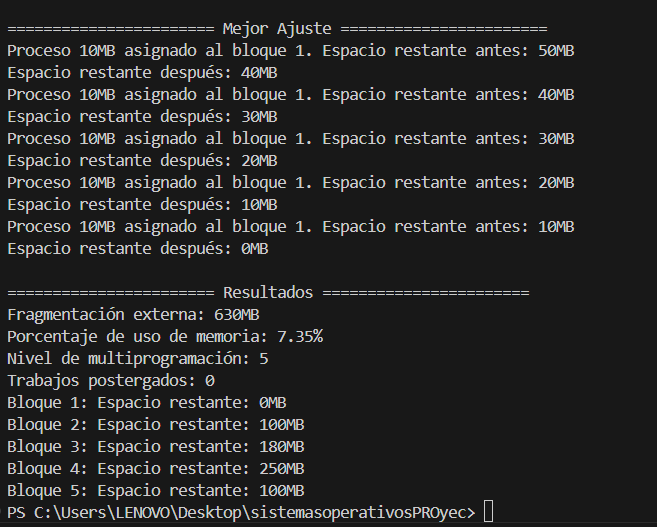


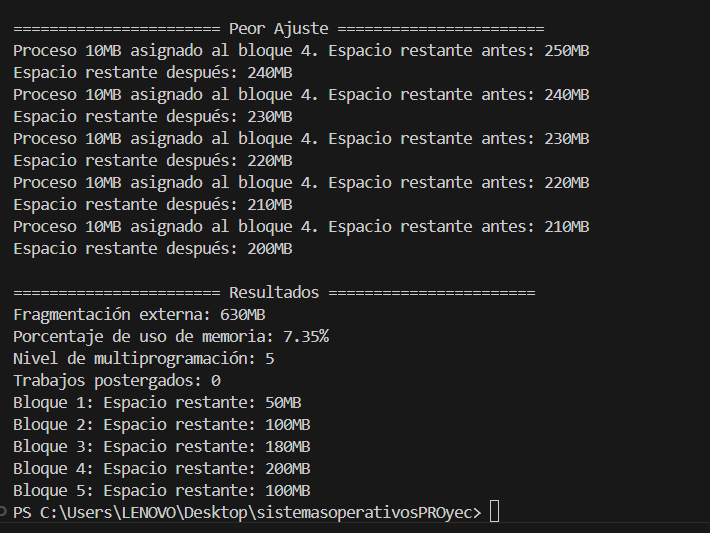
-Partición Fija:

-Primer Ajuste

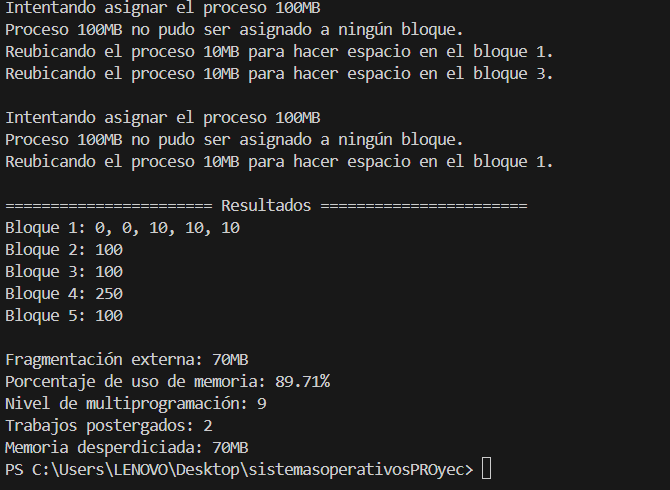


-Mejor Ajuste



-Peor Ajuste

-Particionamiento Reubicable:



**6. Conclusiones y recomendaciones.**

**6.1. Conclusión Gestión de Procesos.**

-FIFO: Este algoritmo es simple y fácil de implementar, pero en escenarios donde existen procesos largos seguidos de procesos cortos, puede generar un problema de convoy effect. Es ideal en sistemas donde la simplicidad es prioritaria y los tiempos de ejecución son similares entre procesos.

-SJN: Ofrece un rendimiento óptimo en términos de tiempo promedio de espera, pero requiere conocer de antemano el tiempo de ejecución de los procesos, lo que no siempre es posible en entornos reales. Es adecuado para sistemas con cargas predecibles y pocos procesos largos.

-Prioridad: Es eficaz para priorizar procesos críticos, pero puede ocasionar inanición de procesos de baja prioridad. El uso de prioridades dinámicas (que aumentan con el tiempo) puede mitigar este problema y equilibrar el sistema.

**6.2. Conclusión Gestión de Memoria.**

-Asignación Contigua Simple: Este método es fácil de implementar, pero tiene una alta fragmentación externa y no aprovecha de manera eficiente la memoria cuando los procesos son de tamaños variables. Es adecuado para sistemas simples o específicos donde los procesos tienen un tamaño constante.

-Partición Fija:

-Primer Ajuste: Es rápido y eficiente para sistemas con una carga de trabajo regular, pero puede generar fragmentación externa si se seleccionan bloques grandes frecuentemente.

-Mejor Ajuste: Reduce la fragmentación externa al usar los bloques más pequeños posibles, pero puede ser más lento debido a la búsqueda exhaustiva del mejor bloque.

-Peor Ajuste: Tiende a dejar grandes bloques de memoria sin utilizar, siendo útil en casos donde se espera la llegada de procesos grandes posteriormente.

-Particionamiento Reubicable: Es la opción más flexible, permitiendo reorganizar la memoria para acomodar nuevos procesos y minimizar la fragmentación externa. Sin embargo, su costo computacional puede ser elevado debido al movimiento constante de procesos en memoria.

**6.3. Recomendaciones.**

Selección de Algoritmo según el Escenario:

Usar FIFO o Primer Ajuste en sistemas simples con baja variabilidad en la carga de trabajo.

Implementar SJN o Mejor Ajuste en sistemas donde se busca eficiencia y se puede predecir el comportamiento de los procesos.

Optar por Prioridad o Particionamiento Reubicable en sistemas críticos que requieran alta flexibilidad y priorización de tareas.

7. Referencia bibliográficas.

- Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). "Operating System Concepts" (10th ed.). John Wiley & Sons.

- Tanenbaum, A. S., & Bos, H. (2015). "Modern Operating Systems" (4th ed.). Pearson.

- Bovet, D. P., & Cesati, M. (2005). "Understanding the Linux Kernel" (3rd ed.). O'Reilly Media.