


# Programa 3 y Reporte 3

# **Simular el Proceso por Lotes**

---

Salvador Castañeda Andrade  
Sistemas Operativos



<b>Introducción.....</b>	<b>2</b>
<b>Desarrollo.....</b>	<b>3</b>
1. Selección del siguiente proceso a ejecutar:.....	3
2. Finalización de procesos:.....	4
<b>Conclusión.....</b>	<b>5</b>
<b>Bibliografía.....</b>	<b>6</b>



## Introducción

Se presenta una simulación de procesamiento por lotes implementada en Python utilizando el algoritmo de planificación FCFS (First Come First Serve). El objetivo de esta simulación es mostrar cómo se pueden gestionar procesos en un sistema informático de manera secuencial, atendiendo a su orden de llegada.

## Desarrollo

### 1. Selección del siguiente proceso a ejecutar:

Python

```
def running_process():

    global pending_processes, current_process, finished_processes,
    current_process_number

    if not current_process and pending_processes:

        current_process = pending_processes.pop(0)

        current_process_number = len(finished_processes) + 1

        current_process.arrival_time = elapsed_seconds # Set the arrival time
        when a process starts

    if current_process:

        if current_process.max_time > 0:

            current_process.max_time -= 1

        else:

            current_process.finish_time = elapsed_seconds # Set the finish
            time when a process finishes

            finished_processes.append(current_process)

            current_process = None

    update_GUI_process_lbl_txt_box()

    lbl_global_clock.after(1000, running_process)
```

En esta función `running_process()`, se verifica si no hay un proceso en ejecución actual (`current_process`) y si hay procesos en espera (`pending_processes`). Si estas condiciones se cumplen, se selecciona el siguiente proceso en la lista de espera (`pending_processes.pop(0)`) y se asigna a `current_process`, lo que refleja el comportamiento FCFS de seleccionar el siguiente proceso en función del orden de llegada.

## 2. Finalización de procesos:

Python

```
if current_process:

    if current_process.max_time > 0:

        current_process.max_time -= 1

    else:

        current_process.finish_time = elapsed_seconds # Set the finish time
        when a process finishes

        finished_processes.append(current_process)

        current_process = None
```

En este fragmento, se comprueba si el proceso actual (`current_process`) ha alcanzado su tiempo máximo de ejecución (`max_time`). Si es así, se marca como finalizado y se agrega a la lista de procesos finalizados (`finished_processes`). Esto sigue el principio FCFS de completar los procesos en el orden en que llegaron.

Estas partes del código ilustran cómo se implementa el algoritmo FCFS en la simulación de procesamiento por lotes.



## Conclusión

La simulación de procesamiento por lotes implementada con el algoritmo FCFS (First Come First Serve) proporciona una visión clara de cómo se pueden gestionar procesos en un sistema informático. A través de la generación aleatoria de procesos, la ejecución secuencial basada en el orden de llegada y la visualización en tiempo real, se demuestra la efectividad y simplicidad de este algoritmo de planificación.

Una de las principales ventajas del algoritmo FCFS es su simplicidad, lo que lo hace fácil de entender e implementar. Además, no requiere una planificación compleja y funciona bien para cargas de trabajo ligeras o moderadas. Sin embargo, presenta algunas limitaciones, como el problema de inanición para procesos de baja prioridad y posibles tiempos de espera largos para procesos que llegan después de una ráfaga de procesos largos.

En conclusión, la simulación de procesamiento por lotes con el algoritmo FCFS ofrece una perspectiva valiosa sobre la gestión de procesos en un sistema informático. A través de esta simulación, se destaca la importancia de elegir el algoritmo de planificación adecuado en función de las necesidades y características específicas del sistema, con el objetivo de optimizar el rendimiento y la eficiencia en la ejecución de procesos.



## Bibliografía

### **Bibliografía básica**

Stallings, W. (2011), Operating Systems: Internals and Design Principles. Prentice Hall. 7th Edition.

Silberschatz, A., Galvin, P. B., & Gagne, G. (2010) Ingeniería del Software: Un enfoque práctico, McGraw Hill, México.

### **Bibliografía complementaria**

Peters, James F. & Pedrycz, Witold (2008) Operating System Concepts. John Wiley & Sons Inc. 8th Edition.

Tanenbaum, A. S. (2008) Modern Operating Systems Pearson Educación, México.

