

---

**Proyecto: Simulador**

Con este proyecto se reforzarán los conocimientos sobre planificación de procesos, algoritmos de scheduling, concurrencia y mecanismos de sincronización (mutex y semáforos). El objetivo es desarrollar una aplicación visual y backend utilizando C o C++, que permita simular visualmente diferentes algoritmos de planificación y escenarios de sincronización.

La aplicación debe constar de las siguientes funcionalidades:

**A. Simulador de Algoritmos de Calendarización:**

**1) Algoritmos soportados:**

- First In First Out
- Shortest Job First
- Shortest Remaining Time
- Round Robin (con quantum configurable)
- Priority (con envejecimiento de procesos)

**2) Carga dinámica de procesos desde archivos .txt:**

- **Procesos:** Este archivo contiene los procesos que ejemplificarán el orden de ejecución según el algoritmo seleccionado.

Formato de archivo (por línea):

<PID>, <BT>, <AT>, <Priority>

**Ejemplo: P1, 8, 0, 1**

**3) Visualización dinámica del scheduling (Requerimientos):**

- Línea de tiempo con bloques representando cada proceso ejecutado en el orden correspondiente según el algoritmo de calendarización seleccionado. (Diagrama de Gantt)
- Número de “ciclo” visible en todo momento. (Asumiendo que los tiempos de ejecución de los Burst Time y los Arrival Time se miden en ciclos)
- Scroll horizontal dinámico cuando la cantidad de eventos exceda el espacio visual disponible.
- Visualización dinámica de la ejecución de los procesos. (Es decir, el diagrama de Gantt no se genera y se pinta sino que la ejecución se va pintando dinámicamente)
- Diferenciación por nombre y por color de cada uno de los procesos.

- Resumen de métricas de eficiencia de los algoritmos seleccionados (Avg Waiting Time y Avg Completion Time).

## **B. Simulador de Mecanismos de Sincronización:**

### **1) Mecanismos soportados:**

- Mutex Locks
- Semáforos

### **2) Carga dinámica desde archivos .txt:**

- **Procesos:** Este archivo contiene los procesos que ejemplificarán el acceso, para lectura o escritura, de los diferentes recursos del sistema.

Formato de archivo (por línea):

<PID>, <BT>, <AT>, <Priority>

**Ejemplo: P1, 8, 0, 1**

- **Recursos:** Este archivo contiene los diferentes recursos disponibles en el sistema simulado.

Formato de archivo (por línea):

<NOMBRE RECURSO>, <CONTADOR>

**Ejemplo: R1, 1**

- **Acciones:** Este archivo contiene las diferentes “acciones” que los procesos realizan sobre los recursos y en qué ciclo durante de la línea de tiempo de la vida del sistema.

Formato de archivo (por línea):

<PID>, <ACCION>, <RECURSO>, <CICLO>

**Ejemplo: P1, READ, R1, 0**

### **3) Visualización dinámica del scheduling (Requerimientos):**

- Línea de tiempo con bloques que representan las diferentes acciones detalladas en el archivo de acciones y reflejando los diferentes estados en los que el proceso puede estar (los estados son: ACCESSED o WAITING dependiendo de si el recurso está disponible o no).
- Número de “ciclo” visible en todo momento. (Asumiendo que los tiempos de acceso a los recursos tanto read como write se miden en ciclos y cada operación dura a lo máximo 1 ciclo)
- Scroll horizontal dinámico cuando la cantidad de eventos exceda el espacio visual disponible.
- Diferenciación visual entre accesos exitosos y esperas.

### **Interfaz Gráfica (GUI) (Requerimientos):**

La interfaz debe permitir fácilmente al usuario realizar lo siguiente:

- Seleccionar tipo de simulación (A. Calendarización o B. Sincronización)
- A. Seleccionar uno o más algoritmos de calendarización y configurar quantum.
- A. Desplegar métricas de eficiencia sobre los procesos cargados.
- B. Seleccionar modo de sincronización (mutex o semáforo).
- Cargar procesos, recursos y acciones desde archivos de texto. (Dependiendo de la simulación)
- Ejecutar la simulación visualmente de manera dinámica.
- Visualizar dinámicamente procesos, estados y acciones en una línea de tiempo gráfica con scroll horizontal.
- Consultar información cargada (procesos, recursos y acciones).
- Limpiar información y reiniciar simulaciones fácilmente.

### **Instrucciones de entrega:**

- El proyecto debe entregarse, a más tardar, el 6 de junio de 2025 a las 18:59 en Canvas, y luego se harán presentaciones en el horario de la clase. Los horarios de presentación serán aleatorios y se realizarán de forma PRESENCIAL.
- El proyecto se debe desarrollar en parejas. Cada pareja producirá un sistema de simulación completo.
- Todas las parejas deben presentar el día que se les asigne y presencialmente. Todo trabajo (código, documentación, README, investigaciones realizadas, etc.) debe estar subido en Canvas como un archivo .zip antes de la fecha y hora de la presentación. No se otorgará calificación a

grupos que no suban su trabajo a Canvas. Solamente es necesario que una persona de la pareja realice la carga de la información.

- Importante agregar un archivo README bien redactado a la documentación de su proyecto con instrucciones de ejecución del simulador y formatos claros para la carga de archivos.

## Evaluación

### Rúbrica

| Componente                    |   | %          |
|-------------------------------|---|------------|
| Simulación de Calendarización | Simulación visual dinámica                                    | 10         |
|                               | Implementación funcional de los algoritmos de calendarización | 35         |
|                               | Carga de procesos y configuración de hiperparámetros          | 5          |
| Simulación de Sincronización  | Simulación visual dinámica                                    | 10         |
|                               | Implementación funcional de los mecanismos de sincronización  | 35         |
|                               | Carga de procesos, recursos y acciones                        | 5          |
| <b>TOTAL</b>                  |   | <b>100</b> |

### Factores de deducción:

- Interfaz desordenada o poco intuitiva (-20%)
- Resolución de fallas de funcionamiento durante presentación. (-5%)
- Presentación deficiente o poco preparada (-5% a -15%)
- Funcionamiento incorrecto con carga dinámica de procesos, recursos y acciones (-30%)
- Manejo de programación defensiva (-20%)
- Presentación de manera remota (-100%)