UNIVERSIDAD NACIONAL JORGE BASADRE GROHMANN

FACULTAD DE INGENIERÍA

ESCUELA PROFESIONAL DE INGENIERÍA EN INFORMÁTICA Y SISTEMAS

GUIA DE USUARO PARA EL SIMULADOR DE SISTEMA OPERATIVO

ASIGNATURA:
Sistemas operativos

SECCIÓN:
Grupo A

Estudiante:
Maykol David Espinoza Kquerare 2025-11902c
Luz Marina Flores Carbajal 2025-11901c

FECHA:
Tacna, diciembre del 2025

1. Requisitos del sistema

El simulador está diseñado para ejecutarse en entornos Linux (Ubuntu/Debian) y requiere el siguiente software:

- **Compilador GCC (GNU Compiler Collection):** Para compilar el código fuente en C.
- **Librería Jansson:** Una librería de C para la manipulación de datos JSON.

1.1. Instalación de requisitos (Ubuntu)

Ejecute los siguientes comandos en su terminal:

 # 1. Actualizar la lista de paquetes

```
sudo apt update
```

 # 2. Instale el compilador GCC y la librería Jansson para headers y binarios

```
sudo apt install build-essential libjansson-dev
```

2. Compilación del Programa

La compilación convierte sus archivos de código fuente (.c y .h) en un único archivo ejecutable (sim_so).

2.1. Estructura de Archivos

Asegúrese de que todos los archivos fuente estén en el mismo directorio (en este caso sera ~/Escritorio/src/):

- main.c
- simulador.c
- simulador.h
- simulador_colas.c
- simulador_memoria.c

2.2. Proceso de Compilación

1. Abra su terminal y navegue al directorio del proyecto:

```
cd ~/Escritorio/src
```

2. Ejecute el comando de compilación. Este comando compila y enlaza todos los archivos, nombrando el ejecutable final como sim_so:

```
gcc main.c simulador.c simulador_colas.c simulador_memoria.c -o sim_so -std=c99 -ljansson
```

3. **Verificación:** Si la compilación fue exitosa, no verá ningún mensaje de error y aparecerá un nuevo archivo llamado sim_so en su directorio.

3. Preparación del Archivo de Configuración

El simulador es dirigido enteramente por un archivo de texto en formato **JSON** el cual de preferencia colocaremos en una ruta relativa llamada config dentro del mismo directorio que los archivos fuentes (config/config.json). Este archivo debe estar en el mismo directorio que el ejecutable sim_so.

3.1. Estructura Esencial del JSON

El archivo debe contener los siguientes cuatro objetos raíz, tal como se muestra en la estructura a continuación:

```
{
    "cpu": { ... },
    "memoria": { ... },
    "procesos": [ ... ],
    "solicitudes_mem": [ ... ]
}
```

3.2. Configuración de CPU ("cpu")

Campo	Función	Valores válidos	Estado

"algoritmo"	Define la política de planificación.	"FCFS", "SPN", "Round Robin"	Obligatorio.
"quantum"	Intervalo de tiempo para la expropiación.	Números Enteros por ejemplo 4.	Solo relevante si "algoritmo" es "RR (Round Robin)".

3.3. Configuración de Memoria ("memoria")

Campo	Función	Valores Válidos	Nota
"tam"	Tamaño total de la memoria principal.	Entero (en bytes, ej: 1048576 para 1 MiB)	Obligatorio.
"estrategia"	Algoritmo de asignación de bloques.	"first-fit", "best-fit"	Obligatorio.

3.4. Definición de Procesos (“procesos”)

Este sera array que lista todos los procesos que entrarán al sistema.

```
// Ejemplo de “procesos”
{
   "pid": 101,          // Identificador (debe ser único)
   "llegada": 5,        // Tiempo en el que el proceso entra a la cola de listos
   "servicio": 25       // Tiempo total de CPU requerido
}
```

3.5. Solicitudes de Memoria (“solicitudes_mem”)

Este es un array que mapea las solicitudes de memoria a los procesos definidos. Un proceso que no está en esta lista asume una solicitud de memoria de 0 bytes.

```
// Ejemplo de “solicitudes_mem”
{
    "pid": 101,          // PID del proceso solicitante (debe existir en "procesos")
    "tam": 150000        // Tamaño de la memoria solicitada en bytes
}
```

4. Ejecución de la Simulación

Una vez que el programa está compilado y su archivo JSON está listo, ejecute el simulador pasándole el nombre del archivo de configuración.

4.1. Comando de Ejecución

```
./sim_so config/config.json
```

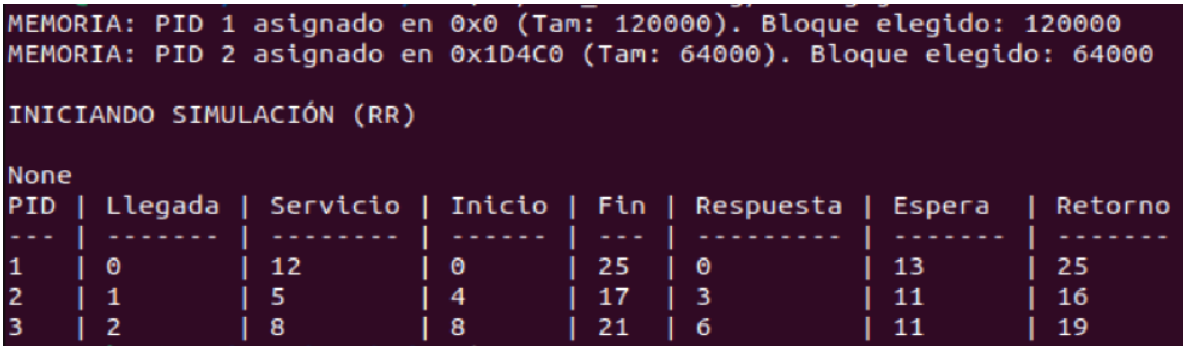
4.2. Interpretación de la Salida

La simulación generará una única tabla de resultados con las métricas clave, ordenada por PID para facilitar el análisis.

Columna	Significado	Análisis
Inicio	Tiempo de la primera ejecución.	Necesario para calcular la respuesta.
Fin	Tiempo en el que el proceso terminó por completo.	Necesario para calcular el retorno.
Respuesta	Retraso entre la llegada y la primera ejecución.	Debe ser bajo para sistemas interactivos.

Espera	Tiempo total que el proceso pasó en la Cola de Listos.	Mide la ineficiencia (tiempo sin hacer trabajo).
Retorno	Tiempo total de vida del proceso en el sistema.	Mide la eficiencia general del sistema.

Formato de salida



```
MEMORIA: PID 1 asignado en 0x0 (Tam: 120000). Bloque elegido: 120000
MEMORIA: PID 2 asignado en 0x1D4C0 (Tam: 64000). Bloque elegido: 64000

INICIANDO SIMULACIÓN (RR)

None
PID | Llegada | Servicio | Inicio | Fin | Respuesta | Espera  | Retorno
--- | ------- | -------- | ------ | --- | --------- | ------- | -------
1   | 0       | 12       | 0      | 25  | 0         | 13      | 25
2   | 1       | 5        | 4      | 17  | 3         | 11      | 16
3   | 2       | 8        | 8      | 21  | 6         | 11      | 19
```