Manual de Usuario

Simulador de Gestor de Procesos para Sistemas Operativos

Universidad Autónoma de Tamaulipas
Materia: Sistemas Operativos
Profesor: Dante Adolfo Muñoz Quintero
Semestre: 7º

Autores:

- Solbes Davalos Rodrigo
- Izaguirre Cortes Emanuel
- Rosales Pereles Denisse Ariadna
- Morales Urrutia Javier Antonio
- Reyes Alejo Emiliano

Introducción

¿Qué es el Simulador de Gestor de Procesos?

El Simulador de Gestor de Procesos es una herramienta educativa diseñada para comprender los conceptos fundamentales de sistemas operativos mediante la simulación interactiva de:

- Gestión de procesos y sus estados
- Algoritmos de planificación de CPU
- Asignación y liberación de recursos
- Comunicación entre procesos
- Sincronización mediante semáforos
- Prevención de deadlocks

Objetivo

Proporcionar una plataforma práctica para experimentar con conceptos teóricos de sistemas operativos sin la complejidad de un sistema real.

Requisitos del Sistema

Requisitos Mínimos

- **Sistema Operativo:** Windows, Linux o macOS
- **Compilador:** GCC (GNU Compiler Collection)
- **RAM:** 512 MB
- **Espacio en Disco:** 10 MB
- **Terminal/Consola:** Cualquier terminal que soporte entrada/salida estándar

Software Necesario

Windows

- MinGW-w64 o Cygwin (para GCC)
- Command Prompt o PowerShell

Linux

- GCC (generalmente preinstalado)

```
sudo apt-get install build-essential  # Ubuntu/Debian
sudo yum install gcc                  # CentOS/RHEL
```

macOS

- Xcode Command Line Tools

```
xcode-select --install
```

Instalación

Paso 1: Obtener los Archivos

Asegúrate de tener los siguientes archivos en un directorio:

- simulador_procesos.c (código fuente)

- Makefile (opcional, para compilación simplificada)
- README.md (documentación)

Paso 2: Compilación

Opción A: Usando Makefile (Recomendado)

```
# Compilar el programa
make

# O compilar con optimización
make release
```

Opción B: Compilación Manual

```
# Linux/macOS
gcc -Wall -Wextra -std=c99 -g simulador_procesos.c -o simulador_procesos

# Windows (con MinGW)
gcc -Wall -Wextra -std=c99 -g simulador_procesos.c -o simulador_procesos.exe
```

Paso 3: Verificar la Instalación

```
# Linux/macOS
./simulador_procesos

# Windows
simulador_procesos.exe
```

Si la compilación fue exitosa, verás la pantalla de bienvenida del simulador.

Inicio Rápido

Primera Ejecución

1. **Ejecutar el programa:**

2. ./simulador_procesos
3. **Seleccionar algoritmo de planificación:**
4. 1. FCFS (First Come First Served)
5. 2. Round Robin
6. Seleccione un algoritmo (1-2): 2
7. **Si eligió Round Robin, ingrese el quantum:**
8. Ingrese el quantum: 3
9. **Accederá al menú principal:**
10. ==========
11. SIMULADOR DE GESTOR DE PROCESOS - MENU PRINCIPAL
12. ==========
13. 1. Crear proceso
14. 2. Listar procesos activos
15. 3. Mostrar recursos disponibles
16. 4. Ejecutar siguiente paso
17. ...

Ejemplo Básico: Crear y Ejecutar un Proceso

Paso 1: Seleccione opción 1 (Crear proceso)

Tiempo de ejecución (burst time): 5

Prioridad (mayor número = mayor prioridad): 3

Bloques de memoria necesarios: 1

Paso 2: Seleccione opción 4 (Ejecutar siguiente paso)

Repita varias veces para ver la ejecución

Paso 3: Seleccione opción 2 (Listar procesos activos)

Para ver el estado de los procesos

Paso 4: Seleccione opción 9 (Ver estadísticas)

Para ver el rendimiento del sistema

Guía de Funcionalidades

1. Crear Proceso

Opción del Menú: 1

Descripción: Crea un nuevo proceso en el sistema con parámetros personalizados.

Parámetros:

Parámetro	Descripción	Valores
Burst Time	Tiempo total de ejecución del proceso	1-999 unidades
Prioridad	Importancia del proceso (mayor = más prioritario)	1-10
Memoria	Bloques de memoria necesarios (1 bloque = 1024MB)	1-4 bloques

Ejemplo:

--- Crear Proceso ---

Tiempo de ejecución (burst time): 10

Prioridad (mayor número = mayor prioridad): 5

Bloques de memoria necesarios (1 bloque = 1024MB, max 4 bloques): 2

✓ Proceso creado exitosamente (PID: 1)

Notas:

- El PID se asigna automáticamente
- El proceso inicia en estado READY
- La memoria se asigna cuando el proceso comienza a ejecutarse

2. Listar Procesos Activos

Opción del Menú: 2

Descripción: Muestra todos los procesos en el sistema con su información detallada.

Información Mostrada:

- PID: Identificador único del proceso
- Estado: LISTO, EJECUTANDO, ESPERANDO, TERMINADO
- Prioridad: Nivel de prioridad del proceso
- Burst: Tiempo total de ejecución
- Restante: Tiempo que falta por ejecutar
- Memoria: Memoria asignada en MB
- Espera: Tiempo acumulado esperando

Ejemplo de Salida:

========== LISTA DE PROCESOS ==========

PID Estado Prior. Burst Restante Memoria Espera

---

1 EJECUTANDO 5 10 7 2048MB 3

2 LISTO 3 6 6 1024MB 5

3 ESPERANDO 7 8 8 3072MB 2

==========

3. Mostrar Recursos Disponibles

Opción del Menú: 3

Descripción: Visualiza el estado actual de los recursos del sistema.

Recursos del Sistema:

Recurso	Total	Descripción
CPU	1 unidad	Procesador del sistema
Memoria	4 bloques (4096 MB)	Memoria RAM disponible

Ejemplo de Salida:

========== RECURSOS DEL SISTEMA ==========

CPU:

Total: 1

Disponible: 0

En uso: 1

Memoria:

Total: 4 bloques (4096 MB)

Disponible: 1 bloques (1024 MB)

En uso: 3 bloques (3072 MB)

==========

4. Ejecutar Siguiente Paso

Opción del Menú: 4

Descripción: Avanza la simulación en 1 unidad de tiempo.

Acciones Realizadas:

1. Incrementa el reloj del sistema
2. Ejecuta el proceso actual (si hay uno)
3. Actualiza tiempos de espera
4. Selecciona el siguiente proceso según el algoritmo
5. Registra eventos en el log

Ejemplo de Salida:

=== Tiempo: 5 ===

CPU: Ejecutando PID 1

¿Cuándo usar esta opción?

- Para simular paso a paso la ejecución
- Para observar cambios de estado
- Para ver cómo trabaja el planificador
- Para demostrar el comportamiento del sistema

5. Suspender Proceso

Opción del Menú: 5

Descripción: Pausa temporalmente un proceso, cambiándolo a estado WAITING.

Uso:

--- Suspender Proceso ---

PID del proceso a suspender: 1

✓ Proceso PID 1 suspendido

Efectos:

- El proceso deja de ejecutarse
- Los recursos permanecen asignados
- El proceso puede reanudarse posteriormente

6. Reanudar Proceso

Opción del Menú: 6

Descripción: Reactiva un proceso suspendido, cambiándolo a estado READY.

Uso:

--- Reanudar Proceso ---

PID del proceso a reanudar: 1

✓ Proceso PID 1 reanudado

Requisitos:

- El proceso debe estar en estado WAITING
- El proceso no debe estar terminado

7. Terminar Proceso

Opción del Menú: 7

Descripción: Finaliza forzadamente un proceso, liberando sus recursos.

Uso:

--- Terminar Proceso ---

PID del proceso a terminar: 1

✓ Proceso PID 1 terminado

Efectos:

- Libera CPU y memoria asignadas
- Calcula estadísticas finales
- Registra causa de terminación (Usuario)

8. Ver Logs

Opción del Menú: 8

Descripción: Muestra el historial de eventos del sistema.

Tipos de Eventos Registrados:

- Creación de procesos
- Cambios de estado
- Asignación/liberación de recursos
- Envío/recepción de mensajes

- Operaciones sobre semáforos
- Terminación de procesos

Ejemplo de Salida:

========== LOGS DEL SISTEMA ==========

Tiempo PID Evento

---

0 -1 Sistema inicializado

0 -1 Algoritmo Round Robin seleccionado (Quantum: 3)

1 1 Proceso PID 1 creado (Burst: 10, Prioridad: 5, Memoria: 2 bloques)

2 1 Recursos asignados a PID 1 (CPU: 1, Memoria: 2 bloques = 2048MB)

3 1 PID 1 ejecutando (Tiempo restante: 9)

==========

Nota: Solo se muestran los últimos 20 eventos. El sistema mantiene hasta 1000 entradas.

9. Ver Estadísticas

Opción del Menú: 9

Descripción: Muestra métricas de rendimiento del sistema.

Métricas Calculadas:

Métrica	**Descripción**	**Fórmula**
Tiempo promedio de espera	Promedio del tiempo que los procesos esperaron	Σ(tiempo_espera) / procesos_completados
Tiempo promedio de retorno	Tiempo desde llegada hasta finalización	Σ(turnaround_time) / procesos_completados
Utilización de CPU	Porcentaje de tiempo que la CPU estuvo ocupada	(tiempo_ocupado / tiempo_total) × 100
Throughput	Procesos completados por unidad de tiempo	procesos_completados / tiempo_total

Ejemplo de Salida:

========== ESTADISTICAS DEL SISTEMA ==========

Tiempo total de simulación: 45 unidades

Procesos completados: 3

Tiempo promedio de espera: 8.33 unidades

Tiempo promedio de retorno: 18.67 unidades

Utilización de CPU: 88.89%

Throughput: 0.0667 procesos/unidad de tiempo

Algoritmo de planificación: Round Robin (Quantum: 3)

==========

10. Enviar Mensaje

Opción del Menú: 10

Descripción: Implementa comunicación entre procesos mediante paso de mensajes.

Uso:

--- Enviar Mensaje ---

PID emisor: 1

PID receptor: 2

Contenido del mensaje: Hola desde proceso 1

✓ Mensaje enviado

Características:

- Comunicación asíncrona
- Mensajes de hasta 256 caracteres
- Sistema de entrega confirmada

- Límite de 100 mensajes en el sistema

11. Recibir Mensajes

Opción del Menú: 11

Descripción: Permite a un proceso recibir los mensajes enviados a él.

Uso:

--- Recibir Mensajes ---

PID receptor: 2

=== Mensajes para PID 2 ===

De PID 1: Hola desde proceso 1

==========

Notas:

- Los mensajes se marcan como entregados
- Solo se muestran mensajes no leídos
- Si no hay mensajes, se indica al usuario

12. Crear Semáforo

Opción del Menú: 12

Descripción: Crea un semáforo para sincronización de procesos.

Uso:

--- Crear Semáforo ---

Valor inicial: 1

✓ Semáforo creado con ID: 0

Parámetros:

- **Valor inicial:** Número de recursos disponibles

- Valor > 0: Recursos disponibles
- Valor = 0: Sin recursos
- Valor < 0: No recomendado

Casos de Uso Comunes:

- Mutex (valor inicial = 1)
- Contador de recursos (valor = número de recursos)
- Barrera de sincronización (valor = 0)

13. Wait en Semáforo

Opción del Menú: 13

Descripción: Operación wait (P) sobre un semáforo.

Uso:

--- Wait en Semáforo ---

ID del semáforo: 0

PID del proceso: 1

✓ Wait ejecutado

Comportamiento:

1. Decrementa el valor del semáforo
2. Si el valor resulta negativo:
    - El proceso se bloquea (estado WAITING)
    - Se añade a la cola del semáforo
3. Si el valor es ≥ 0:
    - El proceso continúa normalmente

14. Signal en Semáforo

Opción del Menú: 14

Descripción: Operación signal (V) sobre un semáforo.

Uso:

--- Signal en Semáforo ---

ID del semáforo: 0

✓ Signal ejecutado

Comportamiento:

1. Incrementa el valor del semáforo
2. Si hay procesos esperando:
    - Desbloquea el primer proceso de la cola
    - Lo cambia a estado READY

15. Demostración Productor-Consumidor

Opción del Menú: 15

Descripción: Configura automáticamente el problema clásico productor-consumidor.

Configuración Automática:

- 2 procesos productores
- 2 procesos consumidores
- Buffer compartido de tamaño 5
- 3 semáforos:
    - mutex (ID: 0, valor inicial: 1) - Control de acceso
    - empty (ID: 1, valor inicial: 5) - Espacios vacíos
    - full (ID: 2, valor inicial: 0) - Espacios llenos

Ejemplo de Salida:

========== DEMOSTRACIÓN PRODUCTOR-CONSUMIDOR ==========

Configuración: 2 Productores, 2 Consumidores, Buffer de tamaño 5

Semáforos creados:

- mutex (ID: 0): control de acceso al buffer
- empty (ID: 1): espacios vacíos en buffer
- full (ID: 2): espacios llenos en buffer

Productor 1 creado (PID: 1)

Productor 2 creado (PID: 2)

Consumidor 1 creado (PID: 3)

Consumidor 2 creado (PID: 4)

Los procesos productores y consumidores han sido creados.

Use 'Ejecutar siguiente paso' para simular la producción/consumo.

==========

Cómo Usarlo:

1. Seleccione la opción 15
2. Use la opción 4 repetidamente para ejecutar pasos
3. Observe los logs (opción 8) para ver la interacción
4. Verifique el estado con la opción 2

Casos de Uso

Caso 1: Comparación de Algoritmos FCFS vs Round Robin

Objetivo: Entender las diferencias entre algoritmos de planificación.

Escenario A: FCFS

1. Iniciar simulador, seleccionar FCFS
2. Crear tres procesos:
    - Proceso 1: Burst=10, Prioridad=5, Memoria=1

    - Proceso 2: Burst=5, Prioridad=3, Memoria=1
    - Proceso 3: Burst=8, Prioridad=7, Memoria=1
3. Ejecutar hasta completar todos
4. Ver estadísticas

Resultados Esperados:

- Los procesos se ejecutan en orden de llegada
- No hay cambios de contexto hasta que termine el proceso actual
- El tiempo de espera puede ser alto para procesos que llegan tarde

Escenario B: Round Robin (Quantum=3)

1. Reiniciar simulador, seleccionar Round Robin con quantum=3
2. Crear los mismos tres procesos
3. Ejecutar hasta completar todos
4. Ver estadísticas

Resultados Esperados:

- Cada proceso ejecuta máximo 3 unidades de tiempo
- Cambios de contexto frecuentes
- Tiempo de espera más equitativo
- Mayor fairness entre procesos

Análisis: Compare los tiempos promedio de espera y retorno entre ambos algoritmos.

Caso 2: Demostración de Deadlock Prevention

Objetivo: Ver cómo el simulador previene situaciones de deadlock.

Pasos:

1. Iniciar simulador
2. Crear proceso 1: Burst=10, Prioridad=5, Memoria=3
3. Crear proceso 2: Burst=8, Prioridad=4, Memoria=2

4. Intentar crear proceso 3: Burst=6, Prioridad=3, Memoria=2

Resultado: El proceso 3 no podrá obtener recursos inmediatamente (solo queda 1 bloque disponible) y pasará a estado WAITING hasta que se liberen recursos.

Observación en Logs:

Solicitud de recursos denegada para PID 3 (prevención deadlock)

PID 3 en espera de recursos

Caso 3: Comunicación Entre Procesos

Objetivo: Simular la comunicación mediante mensajes.

Pasos:

1. Crear dos procesos
2. Desde el proceso 1, enviar mensaje al proceso 2:
    - Emisor: 1
    - Receptor: 2
    - Contenido: "Datos listos para procesar"
3. Proceso 2 recibe el mensaje
4. Proceso 2 envía respuesta al proceso 1
5. Verificar logs para ver el flujo de comunicación

Aplicación Práctica: Este patrón simula comunicación cliente-servidor o pipeline de datos.

Caso 4: Sincronización con Semáforos

Objetivo: Usar semáforos para controlar acceso a recurso compartido.

Escenario: Sección Crítica

1. Crear semáforo mutex con valor inicial 1
2. Crear dos procesos
3. Proceso 1 ejecuta wait en el semáforo (entra a sección crítica)

4. Proceso 2 intenta wait (se bloquea)
5. Proceso 1 ejecuta signal (sale de sección crítica)
6. Proceso 2 se desbloquea y entra a sección crítica

Resultado: Solo un proceso puede estar en la sección crítica a la vez.

Caso 5: Problema Productor-Consumidor Completo

Objetivo: Ver sincronización compleja en acción.

Pasos:

1. Seleccionar opción 15 (Demo Productor-Consumidor)
2. El sistema crea automáticamente productores, consumidores y semáforos
3. Ejecutar múltiples pasos (opción 4)
4. Observar en logs:
    - Productores bloqueándose cuando el buffer está lleno
    - Consumidores bloqueándose cuando el buffer está vacío
    - Exclusión mutua en el acceso al buffer

Verificación:

- Ver logs para confirmar sincronización correcta
- Verificar que no haya condiciones de carrera
- Confirmar que el buffer nunca se sobreescribe

Solución de Problemas

Problema: Error de Compilación

Síntoma:

gcc: command not found

Solución:

- **Linux:** sudo apt-get install gcc

- **macOS:** xcode-select --install
- **Windows:** Instalar MinGW-w64

Problema: El Proceso No Ejecuta

Síntoma: El proceso se queda en estado WAITING permanentemente.

Posibles Causas:

1. No hay recursos disponibles
2. Otro proceso está usando la CPU
3. El proceso está bloqueado en un semáforo

Solución:

- Verificar recursos con opción 3
- Terminar procesos que no necesita
- Ejecutar signal en el semáforo correspondiente

Problema: "Número Máximo de Procesos Alcanzado"

Síntoma: No se pueden crear más procesos.

Causa: Límite de 50 procesos en el sistema.

Solución:

- Terminar procesos completados
- Reiniciar el simulador si es necesario

Problema: Estadísticas Muestran Valores Extraños

Síntoma: Porcentajes mayores a 100% o valores negativos.

Causa: Ningún proceso ha completado su ejecución.

Solución:

- Ejecutar más pasos de simulación

- Asegurarse de que los procesos lleguen a TERMINATED

Preguntas Frecuentes

¿Cuál es la diferencia entre suspender y terminar un proceso?

- **Suspender:** Pausa temporalmente el proceso. Puede reanudarse después. Los recursos permanecen asignados.
- **Terminar:** Finaliza definitivamente el proceso. Libera todos los recursos. No puede reanudarse.

¿Por qué mi proceso está en estado WAITING?

Un proceso puede estar en WAITING por:

1. No hay recursos disponibles (CPU o memoria)
2. Está esperando en un semáforo (bloqueado)
3. Fue suspendido manualmente

¿Cómo funciona el quantum en Round Robin?

El quantum es el tiempo máximo que un proceso puede ejecutar consecutivamente. Cuando se agota:

- El proceso vuelve al final de la cola READY
- Se selecciona el siguiente proceso
- El proceso recupera su quantum completo cuando vuelve a ejecutar

¿Qué pasa si solicito más memoria de la disponible?

El proceso pasa automáticamente a estado WAITING hasta que se libere suficiente memoria. El sistema implementa prevención de deadlock, por lo que no se asignarán recursos si esto podría causar un bloqueo permanente.

¿Puedo tener múltiples semáforos?

Sí, el sistema soporta hasta 10 semáforos simultáneos. Cada uno se identifica con un ID único (0-9).

¿Los mensajes se pierden si el receptor no está listo?

No, los mensajes se almacenan en el sistema hasta que el proceso receptor los lee. Un mensaje solo se marca como entregado cuando el receptor ejecuta la operación de recepción.

¿Qué algoritmo debo elegir?

- **FCFS:** Simple, predecible, sin cambios de contexto. Bueno para procesos largos y sin interacción.
- **Round Robin:** Mejor fairness, ideal para sistemas interactivos. Requiere ajustar el quantum según la carga de trabajo.

¿Cómo interpreto el tiempo de retorno (turnaround)?

El turnaround time es el tiempo total desde que el proceso llega hasta que termina. Incluye:

- Tiempo de espera
- Tiempo de ejecución
- Tiempo bloqueado (si aplica)

Un turnaround bajo indica mejor respuesta del sistema.

Glosario

Término	**Definición**
PCB	Process Control Block - Estructura de datos que contiene información de un proceso
Burst Time	Tiempo total de ejecución que necesita un proceso

Quantum	Unidad de tiempo asignada en Round Robin
Turnaround Time	Tiempo desde llegada hasta finalización de un proceso
Waiting Time	Tiempo que un proceso pasa en cola esperando CPU
Throughput	Número de procesos completados por unidad de tiempo
Deadlock	Situación donde los procesos se bloquean mutuamente esperando recursos
Semáforo	Mecanismo de sincronización para controlar acceso a recursos compartidos
Mutex	Semáforo binario usado para exclusión mutua

Notas Finales

Este simulador es una herramienta educativa. Los valores y comportamientos son simplificaciones de sistemas operativos reales para facilitar el aprendizaje de conceptos fundamentales.

¡Gracias por usar el Simulador de Gestor de Procesos!