Parcial II (Midterm)

Descripción general

A continuación se presenta una guía teórico-práctica para los estudiantes del curso de sistemas operativos. Esta guía permite explorar y experimentar con conceptos relacionados con memoria, asignación de memoria en sistemas paginados y aplicar los principios de la memoria virtual.

Objetivos

Durante el desarrollo de esta guía el estudiante:

- Identificar la memoria física.
- Aplicar los principios de la memoria virtual.
- Recibir solicitudes de asignación y liberación de memoria.
- Aplicar los algoritmos de asignación: First Fit, Best Fit, Worst Fit
- Presentar el estado de la memoria tras cada operación.

Actividad 1. Asignación de memoria

A continuación se describen las tareas a realizar para lograr la comprensión del desafío de programación.

(Tarea 1) Escribe un programa en C++ que lea una secuencia de operaciones desde un archivo de texto o desde consola, con el siguiente formato:

```shell
A <proceso> <tamaño>   // Asignar memoria
L <proceso>            // Liberar memoria
M                      // Mostrar estado de la memoria
```

La memoria total debe ser configurable de al menos 100 unidades. Se deben representar los huecos libres y bloques ocupados. Cada vez que se ejecute M, se debe imprimir un mapa de memoria.

(Tarea 2) Implementar los tres algoritmos de asignación de memoria:

- First Fit: primer bloque libre que quepa.
- Best Fit: bloque libre más pequeño que pueda contener el proceso.
- Worst Fit: bloque libre más grande disponible.

Realizar la asignación de los procesos a los bloques disponibles de la memoria según la política de asignación. Parámetros como tamaño de memoria, algoritmo a utilizar y los archivos de entrada deben ser establecidos a través de la entrada estandar (linea de comandos)

Ejemplo:

```shell
# entrada
A P1 10
A P2 25
L P1
A P3 8
M

# salida
[P2:25][Libre:10][P3:8][Libre:57]
```

(Tarea 3) Incluya en el reporte que entrega la aplicación la fragmentación interna y externa.

Entregable

- Informe de resultados con el análisis comparativo de los algoritmos
- Cree al menos 5 archivos de entrada con asignaciones que usted considere complejas para el algoritmo.
- Video corto sobre la implementación de cada tarea, los hallazgos y las comparaciones

Letra pequeña

- La entrega debe realizarse en las fechas establecidas a través del aula digital. No se recibirán trabajos a través de correo electrónico.
- Agregue al informe los enlaces al código, al video o a cualquier otro recurso que haga parte de la entrega
- La longitud del video debe ser menor a 15 mins.
- Tenga en cuenta los criterios de evaluación descritos en la Rúbrica.
- Solo se reciben archivos en formato PDF a través de la plataforma.
- La implementación de cada tarea/]Actividad debe realizarse en C++

Criterio	1 (Deficiente)	3 (Aceptable)	5 (Excelente)	
Implementación del algoritmo	El algoritmo no corresponde al asignado o no ejecuta correctamente las operaciones básicas de asignación y liberación.	El algoritmo corresponde al asignado y ejecuta correctamente al menos el 70 % de las operaciones, pero presenta errores lógicos o de actualización de memoria.	El algoritmo corresponde exactamente al asignado y ejecuta correctamente todas las operaciones de asignación y liberación sin errores lógicos ni de cálculo.	
Gestión de memoria (asignar y liberar)	Las operaciones de asignación y liberación no actualizan el estado de memoria o generan resultados inconsistentes.	Las operaciones asignan y liberan memoria de forma correcta en la mayoría de los casos, pero pueden fallar ante huecos múltiples o liberaciones consecutivas.	Las operaciones de asignación y liberación actualizan la memoria de forma coherente en todos los casos, incluyendo huecos contiguos, fragmentación y reutilización de espacio.	
Visualización del estado de la memoria	No se muestra el estado de la memoria o el formato es incorrecto (no distingue bloques libres y ocupados).	Se muestra el estado de la memoria, pero con formato incompleto, errores en los tamaños o sin indicar claramente las secciones libres.	Muestra siempre el estado de la memoria en el formato [Proceso: tamaño][Libre: tamaño], actualizándose correctamente después de cada operación.	
Estructura, documentación y claridad del código	Código sin estructura modular, sin comentarios y con nombres de variables no descriptivos.	Código parcialmente modular, con algunos comentarios o estructura limitada.	Código completamente modular (funciones o clases), con nombres descriptivos, comentarios adecuados y formato consistente.	
Casos de prueba y validación	No se incluyen casos de prueba o los datos de entrada no cubren las operaciones básicas.	Incluye al menos tres casos de prueba con asignación y liberación, pero sin cubrir condiciones límite.	Incluye un conjunto de casos que cubren todas las operaciones (asignar, liberar, mostrar), además de casos límite como falta de espacio o liberación repetida.	
Explicación en video (máx. 3 min)	No presenta video o el contenido no explica la lógica del algoritmo.	Presenta video con descripción general, pero sin detallar entradas, salidas o estructura del código.	Presenta video con explicación clara del algoritmo, descripción del código, ejecución paso a paso y análisis de los resultados.	