Tarea Programada II: Algoritmos de Reemplazo de Páginas

Objetivos

- Poner en práctica el funcionamiento de los algoritmos utilizados para llevar a cabo la tarea de reemplazo de páginas cuando se utiliza memoria virtual.

Instrucciones

Para esta tarea, debe trabajar en un programa que permita calcular la cantidad de fallos de página que se generan dada una secuencia de solicitudes de página y un estado de la memoria. Para ello debe tomar en cuenta lo siguiente:

- El uso de los bits: R, M, V/I
- Contadores (de ser necesario)
- Espacio de Swap
- Cantidad de marcos de página - sino se indica, utilice 4 por defecto
- Estado de la memoria - sino se indica, asuma que todos los marcos se encuentran vacíos
- El algoritmo a utilizar
- Cadena de solicitudes

Dado el algoritmo, su programa debe recorrer la cadena de solicitudes en orden, de manera que si la página a solicitar ya se encuentra cargada en memoria, su programa "lo retorne". En caso contrario, su algoritmo debe tener la capacidad de elegir una página víctima utilizando un algoritmo de paginación. Luego de elegir la página víctima debe sacar la página y agregar la nueva página, actualizando debidamente cada estructura de datos, bit y otros, como corresponda. Este proceso debe repetirse hasta que se haya procesado por completo la cadena de solicitudes.

Tome en cuenta, además, de que en la cadena de solicitudes se va a indicar por medio del símbolo (*) que la ejecución del contenido cargado en memoria va a modificar algo en dicha página. Debe asumir que en ausencia del símbolo (*) no se realizan cambios y por ende la página conserva su estado original

Debe programar la funcionalidad de cada algoritmo de acuerdo con la materia vista en clase y el contenido en el libro, de manera que se utilicen las estructuras de datos, contadores, bits y otros a como se especificó. **NO** es permitido realizar cambios a las estructuras de datos o funcionamiento del algoritmo, pues esto llevaría a un "algoritmo nuevo" y a resultados potencialmente distintos.

En este caso, los algoritmos, cuya funcionalidad debe programar, son:

- FIFO
- Second Chance
- NRU (Enhanced Second Chance)
- LRU
- Clock
- LFU
- MFU

Entonces, dada toda esta información, su programa debe recibir una entrada similar a (en el orden indicado en la lista indicada en las instrucciones):

Entrada completa: 4, [0,5,2,8], FIFO, [2,6,1*,8,2,6,2,0,5*,3,1]
Entrada "incompleta": 0, 0, FIFO, [2,6,1*,8,2,6,2,0,5*,3,1]

Como respuesta, luego del procesamiento de la cadena de solicitudes, su programa debe mostar en consola una salida similar a:

Cantidad de fallos de página: 5

Puntos extra (+10 puntos)

- De manera adicional, puede integrar en su programa una funcionalidad que permita, para dada una entrada, despliegue en pantalla el "ranking" de mejor a peor rendimiento de cada algoritmo para esta entrada, indicando la cantidad de fallos de página que produce cada algoritmo bajo el mismo contexto.

Notas adicionales

- Debe correr en el sistema operativo: Ubuntu o Fedora.
- Trabajo de tipo individual o en parejas.
- Debe trabajar en C o C++
- Asegúrese de que su entrega se compile y se ejecute múltiples veces. **Tenga en cuenta que no se revisarán proyectos que no compilen o funcionen adecuadamente.**

Consideraciones del entregable:

- **Buenas prácticas de código:** Cada función, clase o estructura debe tener comentarios claros que expliquen su propósito, modularidad de funcionalidades y nombres descriptivos.
- **Testeo y resultados**: incluir pruebas de su sistema, mostrando cómo se utiliza el programa y los resultados obtenidos.
- **Imágenes de resultados**: se deben de agregar imágenes o screenshots de las salidas ideales o esperables al probar su proyecto
- **README**: se debe de incluir un README que explique cómo compilar, ejecutar el programa y corroborar los test previamente realizados, además de dar una visión general del funcionamiento del programa. Además, debe incluir el carnet de cada uno de los integrantes de la pareja. Éste documento será su documentación externa, asegúrese de entregarlo completo y claro.
- **Formato de entrega:** Toda entrega se debe de hacer en mediación, con un formato de archivo **carnet_TPI.zip** o **carnet&carnet_TPI.zip** que sea la versión final y que incluya lo suficiente para que su código pueda ser correctamente ejecutado.

Entregables:

- Implementación (código y README): 21 de Mayo, 11:59 PM