Campus Monterrey

Escuela de ingeniería y ciencias

TC1031.602 - Programación de estructuras de datos y algoritmos fundamentales

Reflexión Evidencia 2

Profesor:
David Alonso Cantú Delgado

Fernando Morán Fougerat A01284623

Fecha de entrega 7 de octubre del 2023

Reflexión Evidencia 2

Dentro de este curso, hemos avanzado con los temas de ordenamiento, y comenzamos a ver el uso de la herramienta de listas conectadas (Linked lists). Estas son una manera bastante buena de utilizar una lista, ordenarla y agregar nuevas utilidades que son más complicadas al usar otro tipo de unión, como los vectores.

En el programa actual, trabajé con el código de la evidencia anterior que creaba una lista de Logs, y los ordenaba. En esta entrega, se me pidió ordenarlos ya sea por su dirección ip o por la fecha del registro. Se me hizo complicado el poder encontrar elementos dentro de una clase nodo dentro de la lista, pero una vez que encontré su manera, fue posible implementar las funciones necesarias en los ordenamientos y búsqueda secuencial.

El código más sencillo fue el contador. Investigue un poco sobre maneras de contar directo del archivo.txt, y encontré como podía contar usando mapas todos los ips, una herramienta bastante útil y buena de conocer. Una vez que pude ordenar por fechas, guardé un file de txt y pude implementar un contador que separaba según el mes y año que encontraba.

Para poder definir que hace más eficiente a nuestro programa, hay dos cosas que podemos comparar: su tiempo de ejecución y la cantidad de recursos que usa al correrse. El conocer el método de la BigO Notation nos ayuda a ver de manera concreta su eficacia,comparar a otros, y elegir el mejor mecanismo para la entrega.

Dentro del código entregado, el algoritmo de ordenamiento usado fue de QuickSort el cual usa un ciclo recursivo y tiene una eficacia de O(n log n). Al correr el código, el tiempo

de ejecución es poco tardado considerando la cantidad de elementos que ordena (más de 6000), y es más rápido que uno que fuera secuencial, como es el BubbleSort. En mis estudios, es importante comprender las diferentes maneras en las que puedo optimizar mi código, comparar y optar por el más eficaz siempre. Para la búsqueda de rango, use un código secuencial, el cual tiene una eficacia de O(n), pues se tarda n pasos (hasta encontrar lo buscado).