Reflexión de la Actividad 1.3.0

Cuando se trata de procesar información de algún archivo o de otra fuente, es importante tener en cuenta cómo es que se planea acomodar cada uno de los datos recibidos, para así tener un orden y sea mucho más fácil su manipulación. Existen distintos algoritmos de ordenamiento y cada uno de ellos cuenta con una dificultad diferente, y por esto, es que cada uno tiene un tiempo de ejecución diferente. Entre más sencillo el programa, puede que sea más tardado e ineficiente el proceso de ordenamiento, sin embargo, funcionan mucho para las personas que apenas están comenzando en la programación o para aquellos proyectos en los que no importa mucho la memoria y eficiencia. Del otro lado, un programa más complejo y que tarde menos en ejecutarse, funciona más para personas que se encuentran trabajando en proyectos más avanzados en los que cada pedazo de memoria o cada segundo de ejecución cuenta, intentando reducir estos dos lo más posible.

Los algoritmos de búsqueda y ordenamiento son temas que son fundamentales para un ámbito profesional, ya que ayudan a interpretar información y permiten a los programadores manipular dicha información con mayor facilidad. El utilizar el algoritmo adecuado dependiendo del proyecto es de suma importancia y puede llegar a consagrar a un programador como mejor o más eficiente sobre los demás, dándole así una ventaja competitiva en el mundo laboral. Por lo que, entre un mayor conocimiento tengamos en temas esenciales como lo son este tipo de algoritmos, más podremos resaltar para las empresas en un futuro.

Hablando de la situación problema que nosotros desarrollamos, el uso de estos algoritmos es de suma importancia, ya que básicamente de eso se trató la entrega. Consistía en permitirle al usuario obtener un archivo de todos los registros que se dieran a partir de un momento específico hasta una fecha limite dada por el mismo usuario. Para esto lo que hicimos fue, leer todas las entradas de la bitácora proporcionada e insertarlas en un vector, para así poder realizar un método de ordenamiento de nuestro gusto. En mi caso, yo utilicé *Merge Sort*, un algoritmo de ordenamiento un poco más complicado pero que dado la cantidad de entradas que se debían de leer, realizaba su tarea de ordenamiento de una manera más eficaz. Después, contando ya con los datos ordenados, pudimos utilizar un método de búsqueda para la fecha de inicio y de fin. Para esto utilizamos la búsqueda binaria, la cual es más eficiente que la iterativa y requiere que los datos se encuentren ordenado, por eso la inclusión del *Merge Sort*. Finalmente, solo procedí a obtener todos los datos que se encuentran dentro del rango de estas dos fechas y los inserté en un nuevo archivo.