Gabriel Roque Villasana A01280530

Reflexión 2

Se nos ha traído a nuestra atención el mismo problema que la evidencia 1, pero en este caso, se nos pide que separemos los datos del archivo de texto de diferentes barcos que pasan por el canal de suez, creando dos listas doblementes enlazadas para poner en una los barcos que llegan desde el mar muerto y otra que llegan desde el mar rojo, teniendo que ordenar por UBI y buscar y mostrar la cantidad de barcos que entraron desde el mar rojo y muerto, por mes y año.

El uso de doble lista encadenada fue una manera interesante de ordenar los objetos, con ventajas como insertar y colocar contenido en memoria de manera rápida, además de tener una forma de implementación muy rápida, además de que a diferencia de una lista enlazada, es más fácil borrar elementos, tanto como pasar por ella, pues hay dos formas de pasar por ella, dependiendo a dónde queremos ir. A pesar de estas ventajas, no creo que haya sido la forma más rápida o mejor el haber usado esta lista, ya que a diferencia de otras estructuras como vectores, la lista doblemente enlazada siempre que se quiere acceder a un valor, se tiene una complejidad O(n), por lo que usar binary search o valores que piden el uso de muchos valores entre cabeza o cola, requieren más tiempo que otras estructuras que lo realizan en complejidad O(1), además de que comparada con un arreglo toma más espacio.

La forma para mostrar los objetos fue usando 2 diferentes while loops dentro de un while, permitiendo que se tarde con una complejidad máxima de O(n^2), el uso de secuencial search y quicksort hacen que se tenga una complejidad máxima de O(n) y O(n^2) respectivamente. Aunque es sencillo implementar una lista doblemente enlazada, no recomendaría trabajar con ella en estos casos, pues la velocidad es importante para encontrar un valor, y no se puede encontrar más rápidamente que secuencial.