Emilia Victoria Jácome Iñiguez.

A00828347 - ITD

Reflexión Personal:

Tras efectuar esta actividad me di cuenta de la importancia y las ventajas de utilizar distintas estructuras de datos lineales tales como LinkedLists, Queues y Stacks. En primera instancia, una Linked List es una estructura de datos que hace uso de la memoria dinámica. Es decir, los elementos pertenecientes a esta lista no están almacenados en espacios de memoria contiguos. En lugar de utilizar memoria local, los elementos están vinculados mediante apuntadores que van desde el primer elemento al siguiente en un solo sentido. Toda Linked List empieza con un apuntador clave a su primer elemento al que usualmente se denomina HEAD mientras que el último elemento siempre apuntará a NULL (nada). Cada uno de los elementos que componen la Linked List se lo llama Nodo y dentro de sus atributos se podrá almacenar el dato exacto al que se esta referenciando y un apuntador a su elemento próximo. Dentro de las ventajas de utilizar una LinkedList en lugar de un arreglo, la principal y más importante es que no consume espacio de memoria innecesario lo que se traduce en un mayor nivel de eficiencia en el CPU y menos tiempo de compilación del código independientemente de la cantidad de datos con la que se quiera almacenar. Dado que los espacios de memoria se utilizarán en el tiempo de ejecución del programa. Esto es de suma importancia al momento en el que se está lidiando con una gran cantidad de datos. Por otro lado, está la ventaja de que puede crecer o decrecer de tamaño conforme a las necesidades del usuario por lo que los métodos de inserción y borrado de un dato particular es bastante fácil. (geeksforgeeks) Cabe recalcar que para esta actividad en particular el uso de una variante de Linked List, me permitió fácilmente incorporar otras estructuras de datos lineales similares como fue la Queue, A diferencia de la anterior entrega de evidencia se leyeron los datos del archivo y se los almacenó dentro de una Double Linked List la cual es precisamente una Linked List con la diferencia que cuenta además con un apuntador al último elemento (TAIL), por ende, tiene muchas más funcionalidades ya que cada elemento cuenta con dos apuntadores diferentes, uno apuntando al siguiente elemento y el otro al elemento previo por lo que fue mucho más fácil acceder a los últimos elementos sin necesidad de recorrer todo el arreglo partiendo desde HEAD lo que me garantiza un alto nivel de eficiencia en mi programa.

Similarmente, una Queue es una estructura de datos lineal que sigue un orden particular para realizar sus operaciones. Esto significa que es mucho menos flexible que una LinkedList dado que todos sus elementos solo podrán ser agregados en un orden lineal. Una analogía importante para modelar este tipo de estructura corresponde a una fila de supermercado en donde el primer cliente que llega, será el primero en salir y el último en llegar será el último en ser atendido. De esta forma, el primer nodo que ingresa a un Queue será el head y conforme se vayan agregando más elementos, se irán apuntado con respecto al primer elemento. Dentro de las ventajas de utilizar Queues es la efectividad e intercomunicación de los elementos como un proceso interno de la computadora y que se mantiene el mismo orden en toda la lista lo que tiene una aplicación importante en los servicios de sistemas de atención al cliente. Al integrar una Queue dentro de una Double Linked List me permitió tener un acceso más directo a sus elementos e ir vaciando la lista conforme iba utilizando dichos elementos. Precisamente el uso de una Double Linked List me forzó a crear nodos con dos apuntadores cada uno, lo cuál me permitió crear a su vez una Double Queue la cuál tendría las propiedades de una Queue regular en dos ordenes diferentes. (geeksforgeeks)

En otro aspecto, el uso de templates le otorga universalidad a mi programa de modo que podré reutilizar estas estructuras y clases y adaptar sus funcionalidades para cualquier tipo de dato aparte de mi

estructura denominada Dato que fue específicamente diseñado para almacenar cada uno de los campos del archivo de fechas. Otra ventaja de utilizar estructuras lineales corresponde al nivel de organización del programa en general dentro de distintos scopes. A diferencia de mi primera evidencia en la que se utilizó vectores para almacenar los datos, todos los métodos específicos de cada estructura estaban en su debido header y no en el main principal. No obstante, en esta actividad el main contiene precisamente y muy resumidamente los aspectos que son específicos a las propiedades de este problema y enlista muy sencillamente las acciones a realizar. De modo que cuando otra persona lo lea, podrá entender muy fácilmente la sucesión de las acciones básica de las especificaciones del problema sin recurrir a los headers donde ya se da el proceso interno de mayor complejidad. En adición, la cantidad de funciones que tenía en el main, se redujo al igual que la cantidad de variables globales, lo cual es un indicador clave de menos uso de memoria estática.

Referencias:

geeksforgeeks. (s.f.). *Linked List Data Structure*. Recuperado el 7 de Octubre de 2020, de geeksforgeeks.org/: https://www.geeksforgeeks.org/data-structures/linked-list/

geeksforgeeks.org. (7 de Octubre de 2020). *geeksforgeeks.org*. Obtenido de https://www.geeksforgeeks.org/queue-data-structure/