Adrian Gonzalez Saldivar
A00827845
25/10/2020

Act 3.4 - Actividad Integral de BST (Evidencia Competencia)

Un ABB (arbol de busqueda binaria) tiene varias ventajas en su implementación, una de ellas es su eficacia en la memoria, ya que esta no utiliza más memoria de lo que es necesaria. En un ABB todos los elementos del subárbol del lado izquierdo son menores que los elementos almacenados en el subárbol derecho son mayores. Es posible realizar esto ya que al insertar un nuevo dato este se compara con el dato almacenado en la raíz y/o padre, dependiendo de su nivel, el cual dirigirá su dirección a un subárbol izquierdo o derecho. Esto hace demostración de su habilidad en organizar los datos en una manera que permite ser accesible fácilmente. Otro punto que se debe de mencionar es que en este ABB sus elementos solo son almacenables una vez, por lo tanto no hay repetición de datos. Todo esto permite que la búsqueda de un dato, de un nodo, sea más eficiente, tome menos tiempo.

En esta situación problema fue una ventaja el utilizar un ABB, ya que al acomodar todos los nodos en el orden del que nos era pedido, se nos simplifica al acceder a los datos que se nos pedían imprimir, estos eran los más repetidos que por ende estarían en el subárbol derecho. Igualmente algo importante en el código fue el uso de la librería de queue y struct, el cual el segundo nos permitio guardar dos datos relacionados de diferentes tipos en un mismo nodos y el primero nos ayudó a guardar y sacar los nodos buscados.

Las complejidades de los métodos de un ABB (BST) se basan en la altura del árbol, es decir, se basan en la cantidad de datos ingresados y la cantidad de niveles que contienen los subárboles izquierdo y derecho.