****

**Universidad Estatal a Distancia**

**Diplomado en Informática**

**Cátedra de Tecnología de Sistemas**

**Escuela de Ciencias Exactas y Naturales**

**Proyecto #2**

**Estructuras de Datos**

**Agosto 2015**

Olman A Rojas Espinoza

INTRODUCCION

En el siguiente proyecto vamos analizar cómo es que trabaja un Árbol Binario en el lenguaje de programación JAVA.

En primera instancia cabe de afirmar que este proyecto se realizó únicamente para entrada de datos de tipo enteros, por lo cual lo que se hace es introducir estos datos ordenados previamente.

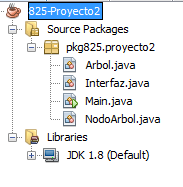
Esto quiere decir que dentro del código existe una manera para discernir si el dato que viene a continuación es mayor o menor que su padre dentro de la estructura del árbol.

Este algoritmo usa fuertemente los métodos recursivos y la creación de nodos personalizados estos a su vez crean una cadena así mismos tipo puntero en la memoria.

Cabe mencionar que el código se encuentra documentado internamente por lo cual interpretarlo es sencillo para cualquier persona que tenga conocimientos en lenguaje de programación JAVA.

DESARROLLO

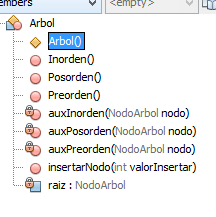
Para empezar este programa sólo cuenta con los siguientes archivos:



Donde

**Arbol.java**

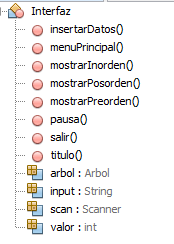
Es la clase que contiene la lógica para estructurar el árbol, además contiene los métodos para imprimir en el orden que se solicita:



**Interfaz.java**

Esta clase contiene todos los métodos para dar una presentación al usuario, está es de tipo consola, valida que los datos sean los correctos y no permite la entrada de datos incorrectos.

Elementos de la clase interfaz.java



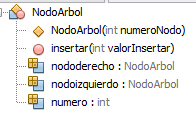
**Main.java**

Es la clase principal que contiene el método public main, lo único que procede es a instanciar a interfaz.java



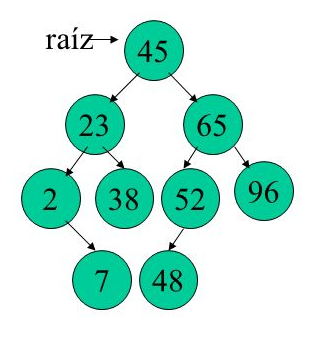
**NodoArbol.java**

Es la clase que crea los nodos, se referencia así misma de forma recursiva y también discierne si el nodo debe ser izquierdo o derecho.

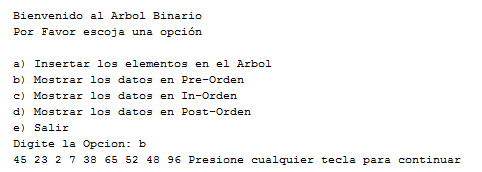


Ejemplo de Funcionamiento:

Se Insertó el siguiente árbol de forma aleatoria, procurando que la raíz fuera 45.



Después de pasar una serie de validaciones como introducción de letras o números negativos, por fin logramos crear el árbol, y procedimos a imprimir en Pre-Orden, a continuación el resultado:



Como se puede apreciar el algoritmo lo hizo correctamente.

CONCLUSION

Como se pudo observar en el programa anterior, la estructura de datos de árbol binario, sirve para hacer una estructura jerárquica ordenada con anticipación, se utilizan nodos para asociar la continuación de la estructura y donde empieza y donde termina.

Trabajar con nodos y punteros de memoria ayuda aprovechar mejor el espacio en memoria ya que se pueden aprovechar aquellos huecos de memoria que no están siendo utilizados ni han sido reclamados por otro proceso.

En comparación con un array el problema surge que el array necesita un espacio de memoria definido y continuo, sin embargo para hacer la ordenación se debe de recorrer varias veces para poder definir un orden.

Contrario a un Árbol Binario porque este se define en su ordenamiento desde un principio, de tal manera que si se quiere encontrar un dato los hilos de procesos se pueden dividir en sus tareas y encontrar la ruta más cercana al nodo que contiene la información.

BIBLIOGRAFIA.

Weiss Mark, A. (2013). *Estructura de Datos en Java.* Madrid España: Pearson Educación S.A.