PRIMERA TAREA PROGRAMACION III

ARBOL BINARIO

DDD

ELMER ISTUSPE RUIZ 7690-21-10969

```
class nodo { // PRIMERA CLASE
   nodo izquierdo; // Tengo mi base y creo una rama llamada izquierda
   nodo derecho;  // Tengo mi base y creo una rama llamada derecha
   int valor;
   public nodo(int va) { // mi constructor
      this.valor = va;
      izquierdo = null; // Le indico que mis ramas tengan un valor nuloo
      derecho = null;
   }
}
public class TareArbol {
   public static void main(String[] args) {
      ArbolBinario arbol = new ArbolBinario();
      arbol.insertar(45);
      arbol.insertar(23);
      arbol.insertar(2); // tomara como raiz 2 por ser el pequeño
      arbol.insertar(7);
      arbol.insertar(38);
      arbol.insertar(65);
      arbol.insertar(52);
      arbol.insertar(96);
      System.out.println("INORDEN");
      arbol.inOrden(arbol.raiz); // es necesario ponderlo, porque es donde se vera los datos
      System.out.println("\n");
      System.out.println("PREORDEN");
      arbol.PreOrden(arbol.raiz);
       System.out.println("\n");
      System.out.println("POSTORDEN");
      arbol.PostOrden(arbol.raiz);
```

```
    }
}
class ArbolBinario { // SEGUNDA CLASE
    nodo raiz;
    public ArbolBinario() {
        raiz = null;
    }
    public void insertar(int valor) { // funcion insertar
        nodo nuevo = new nodo(valor);
        if (raiz == null) {
            raiz = nuevo;
            return;
        }
        nodo root = raiz;
        nodo nodoPadre;
        while (true) {
            nodoPadre = root;
            if (valor < root.valor) {
                root = root.izquierdo;
                if (root == null) {
                    nodoPadre.izquierdo = nuevo;
                    return;
                }
            } else {
                root = root.derecho;
                if (root == null) {
                    nodoPadre.derecho = nuevo;
                    return;
                }
```

```
        }
    }
}
public void inOrden(nodo nodo) { // Aqui se ve como hace el recorrido, si es inorden, postorden, Preorden
    if (nodo != null) {
        inOrden(nodo.izquierdo); // aqui hice el empezara por el izquierdo y solo combino Nodo.izquierdo, Nodo.derecho y mi raiz
        System.out.print(nodo.valor + " ");
        inOrden(nodo.derecho);
    }
} // finalize
public void PreOrden(nodo nodo) { // agrego mi preorden

    if (nodo != null) {
        System.out.print(nodo.valor + " ");
        inOrden(nodo.izquierdo);
        inOrden(nodo.derecho);

    }
}
public void PostOrden(nodo nodo) { // agrege otro que seria postorden
    if (nodo != null) {
        inOrden(nodo.izquierdo);
        inOrden(nodo.derecho);
       System.out.print(nodo.valor + " ");


    }
}
```

}

```
run:
INORDEN
2 7 23 38 45 52 65 96

PREORDEN
45 2 7 23 38 52 65 96

POSTORDEN
2 7 23 38 52 65 96 45 BUILD SUCCESSFUL (total time: 1 second)
```

https://github.com/esrue/TAREAS-PROGRAMACION-3.git