Explique los recorridos PostOrden, PreOrden, InOrden.

- PostOrden: Inicia yendo hacia la hoja izquierda del nodo actual vuelve a aplicar, luego pasa a la hoja derecha vuelve a aplicar si aun no llega al final, por último muestra el dato del nodo en el que este este. En otras palabras muestra las hojas primero y luego la raíz.

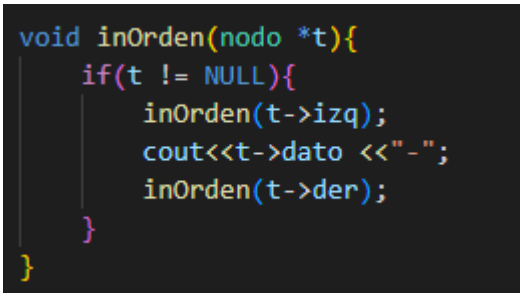
```
//PostOrden
void postOrden(nodo *t){
    if(t != NULL){
        postOrden(t->izq);
        postOrden(t->der);
        cout<<t->dato<<"-";
    }
}
```

- PreOrden: Entra en el nodo actual, para a la hoja de la izquierda, aplica lo primero si no ha llegado al final y por último una vez aplicado pasa a la hoja derecha a aplicar lo anterior. En otras palabras muestra primero la raíz y luego las hojas.

```
void preOrden(nodo *t)
{
    if(t!=NULL)
    {

        cout<<t->dato<<" - ";
        preOrden(t->izq);
        preOrden(t->der);
    }
}
```

- InOrden: Primero se muestra la hoja de la izquierda y vuelve a aplicar, luego imprime el nodo en el que está , luego pasa a la hoja derecha y aplica otra vez. En otras palabras muestra primero una hoja, luego la raíz y luego la otra hoja.



```
void inOrden(nodo *t){
    if(t != NULL){
        inOrden(t->izq);
        cout<<t->dato <<"-";
        inOrden(t->der);
    }
}
```

Debe incluir el código que se subió al aula virtual: Ingresar al árbol, mostrar el árbol

El método inserta un nuevo nodo en un árbol binario.

- Comprueba si el árbol está vacío. Si el árbol está vacío, el nuevo nodo se convierte en la raíz del árbol.

- Si el árbol no está vacío, el método recorre el árbol hasta encontrar un lugar donde insertar el nuevo nodo. El método recorre el árbol de izquierda a derecha, comparando el valor del nuevo nodo con el valor de cada nodo del árbol. Si el valor del nuevo nodo es menor que el valor del nodo actual, el método sigue recorriendo el subárbol izquierdo del nodo actual. Si el valor del nuevo nodo es mayor o igual que el valor del nodo actual, el método sigue recorriendo el subárbol derecho del nodo actual.
- Cuando el método encuentra un nodo que no tiene un hijo del lado correspondiente al valor del nuevo nodo, inserta el nuevo nodo en ese lugar.

método verArbol() funciona de la siguiente manera:

- Si el árbol está vacío, el método no imprime nada.
- Si el árbol no está vacío, el método imprime el contenido del subárbol izquierdo del árbol.
- A continuación, el método imprime el contenido del nodo actual del árbol.
- Por último, el método imprime el contenido del subárbol derecho del árbol.

(Esto esta en el archivo que dice main)

Programar de forma recursiva los recorridos antes planteados

Programar de forma iterativa los recorridos antes planteados