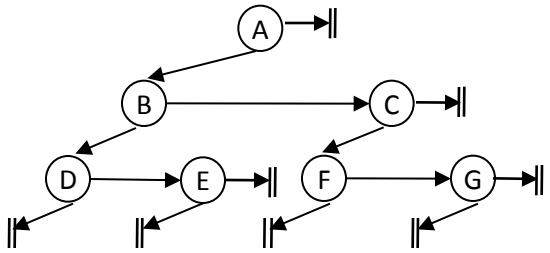
ADT TREE

V javi lahko definiramo abstraktni razred:

```
public abstract class Tree {
    public abstract TreeNode parent(TreeNode n) ;
    public abstract TreeNode leftmostChild(TreeNode n) ;
    public abstract TreeNode rightSibling(TreeNode n) ;
    public abstract void create(Object v, ListLinked subTrees) ;
    public abstract TreeNode root() ;
    public abstract void makenull() ;
    public Object label(TreeNode n) {
       return n.element ;
    }
} // class Tree
```

PRIMER: VIŠINA DREVESA

- RIGHT_SIBLING in LEFTMOST_CHILD omogočata dostop do levega otroka in desnega brata
- Rekurzija gre lahko v ti dve smeri
- Robni primer: prazno poddrevo




```
public int height(TreeNode n) {
   if (n==null)
      return 0 ;
   else
         return Math.max(height(leftmostChild(n))+1,
                         height(rightSibling(n))) ;
} // height
```

Če sta operaciji O(1),
potem računanje višine O(n)

PREMI OBHOD

```
public void preorder(TreeNode r) {
   if (r != null) {
      // najprej izpisemo oznako korena
      r.writeLabel() ; System.out.print(",␣") ;
      TreeNode n = leftmostChild(r) ;
      while (n != null) {
         preorder(n) ;
         n = rightSibling(n) ;
      } // while
   } // if
} // preorder
```

Obratni obhod

```
public void postorder(TreeNode r) {
   if (r != null) {
      TreeNode n = leftmostChild(r) ;
      while (n != null) {
         postorder(n) ;
         n = rightSibling(n) ;
      } // while
      // nazadnje izpisemo oznako korena
      r.writeLabel() ; System.out.print(",␣") ;
   } // if
} // postorder
```

VMESNI OBHOD

```
public void inorder(TreeNode r) {
   if (r != null) {
      TreeNode n = leftmostChild(r) ;
      inorder(n) ;
      // za levim poddrevesom izpisemo oznako korena
      r.writeLabel() ; System.out.print(",␣") ;
      n = rightSibling(n) ;
      while (n != null) {
         inorder(n) ;
         n = rightSibling(n) ;
      } // while
   } // if
} // inorder
```

Obhod po nivojih

Po nivojih izpisujemo tako, da
povečujemo nivo
dokler se še kaj izpiše.

```
public void printByLevel() {
  int l = 1 ;
  while (printLevel(l,root())) {
    l++ ;
    System.out.println() ;
  }
} // printByLevel
```

Rekurzija teče po številu nivoja in po korenu poddrevesa.

Obhod po nivojih

```
private boolean printLevel(int l, TreeNode r) {
    if (r==null)
       return false ;
    else if (l==1) {
       r.writeLabel(); System.out.print(",␣") ;
       return true ;
     }
     else {
       TreeNode n = leftmostChild(r) ;
       boolean existsLevel = false ;
       while (n !=null) {
          // izpis nivoja je mozen v vsaj enem poddrevesu
          existsLevel |= printLevel(l−1, n) ;
          n = rightSibling(n) ;
       }
       return existsLevel ;
    }
} // printLevel
```