

```
// crea un nuovo albero  
// restituisce la radice dell'albero creato
```

```
TREE Tree(ITEM v)  
    TREE t = new TREE  
    t.parent ← nil  
    t.left ← t.right ← nil  
    t.value ← v  
  
    return t
```

```
insertLeft(TREE t)  
    if left ≠ nil then  
        t.parent ← this  
        left ← t
```

```
insertRight(TREE t)  
    if right ≠ nil then  
        t.parent ← this  
        right ← t
```

```
// elimina ricorsivamente il sottoalbero sinistro
```

```
deleteLeft(TREE t)  
    if left ≠ nil then  
        left.deleteLeft  
        left.deleteRight  
        left ← nil
```

```
// elimina ricorsivamente il sottoalbero destro
```

```
deleteRight(TREE t)  
    if right ≠ nil then  
        right.deleteLeft  
        right.deleteRight  
        right ← nil
```