­­­­­­­­­

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра системного програмування і спеціальних комп’ютерних систем

**Лабораторна робота №2**

З дисципліни «Архітектура комп’ютерів - 2»

«Реалізація бінарного дерева. Оптимізація роботи»

**Виконав:**

**студент IV-го курсу**

**групи КВ-41**

**Горпинич-Радуженко Іван**

**Київ 2017**

**Лістинг:**

**package** main  
  
**import** (  
 **"errors"**)  
  
**type** Node **struct** {  
 Value int  
 Left \*Node  
 Right \*Node  
}  
  
**func** (n \*Node) Insert(value int) error {  
  
 **if** n == nil {  
 **return** errors.New(**"Cannot insert a value into a nil tree"**)  
 }  
  
 **switch** {  
 **case** value == n.Value:  
 **return** nil  
 **case** value < n.Value:  
 **if** n.Left == nil {  
 n.Left = &Node{Value: value}  
 **return** nil  
 }  
 **return** n.Left.Insert(value)  
 **case** value > n.Value:  
 **if** n.Right == nil {  
 n.Right = &Node{Value: value}  
 **return** nil  
 }  
 **return** n.Right.Insert(value)  
 }  
 **return** nil  
}  
  
**func** (n \*Node) Find(s int) (bool) {  
  
 **if** n == nil {  
 **return *false*** }  
  
 **switch** {  
 **case** s == n.Value:  
 **return *true* case** s < n.Value:  
 **return** n.Left.Find(s)  
  
 **default**:  
 **return** n.Right.Find(s)  
 }  
}  
  
**func** (n \*Node) findMax(parent \*Node) (\*Node, \*Node) {  
 **if** n.Right == nil {  
 **return** n, parent  
 }  
 **return** n.Right.findMax(n)  
}  
  
**func** (n \*Node) replaceNode(parent, replacement \*Node) error {  
 **if** n == nil {  
 **return** errors.New(**"replaceNode() not allowed on a nil node"**)  
 }  
  
 **if** n == parent.Left {  
 parent.Left = replacement  
 **return** nil  
 }  
 parent.Right = replacement  
 **return** nil  
}  
  
**func** (n \*Node) Delete(s int, parent \*Node) error {  
 **if** n == nil {  
 **return** errors.New(**"Value to be deleted does not exist in the tree"**)  
 }  
  
 **switch** {  
 **case** s < n.Value:  
 **return** n.Left.Delete(s, n)  
 **case** s > n.Value:  
 **return** n.Right.Delete(s, n)  
 **default**:  
 **if** n.Left == nil && n.Right == nil {  
 n.replaceNode(parent, nil)  
 **return** nil  
 }  
  
 **if** n.Left == nil {  
 n.replaceNode(parent, n.Right)  
 **return** nil  
 }  
 **if** n.Right == nil {  
 n.replaceNode(parent, n.Left)  
 **return** nil  
 }  
  
 replacement, replParent := n.Left.findMax(n)  
  
 n.Value = replacement.Value  
  
 **return** replacement.Delete(replacement.Value, replParent)  
 }  
}

**Тестування:**

E:\Projects\Goland\src\BinaryTree1\main>go test

Find node '9999999': Found!

PASS

ok BinaryTree1/main 315.637s