Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

**МЕХАНІКО-ТЕХНОЛОГІЧНИЙ ФАКУЛЬТЕТ**

Кафедра програмування та захисту інформації

Звіт

з виконаної лабораторної роботи № 6

дисципліни “ Технології розробки алгоритмів ”

на тему

“ Дерева. Видалення елементів з бінарного дерева. Невузлове представлення бінарних дерев, збалансовані бінарні дерева, дерево з випадковим пошуком ”

Виконав

студент академічної

групи КІ-15

Аннаев А

Перевірив

Викл.

Гермак В.С.

Кропивницкий- 2017

public void Remove(BinaryTree node)

{

    if(node == null) return;

    var me = MeForParent(node);

    //Если у узла нет дочерних элементов, его можно смело удалять

    if (node.Left == null && node.Right == null)

    {

        if (me == BinSide.Left)

        {

            node.Parent.Left = null;

        }

        else

        {

            node.Parent.Right = null;

        }

        return;

    }

    //Если нет левого дочернего, то правый дочерний становится на место удаляемого

    if (node.Left == null)

    {

        if (me == BinSide.Left)

        {

            node.Parent.Left = node.Right;

        }

        else

        {

            node.Parent.Right = node.Right;

        }

        node.Right.Parent = node.Parent;

        return;

    }

    //Если нет правого дочернего, то левый дочерний становится на место удаляемого

    if (node.Right == null)

    {

        if (me == BinSide.Left)

        {

            node.Parent.Left = node.Left;

        }

        else

        {

            node.Parent.Right = node.Left;

        }

        node.Left.Parent = node.Parent;

        return;

    }

    //Если присутствуют оба дочерних узла

    //то правый ставим на место удаляемого

    //а левый вставляем в правый

    if (me == BinSide.Left)

    {

        node.Parent.Left = node.Right;

    }

    if (me == BinSide.Right)

    {

        node.Parent.Right = node.Right;

    }

    if (me == null)

    {

        var bufLeft = node.Left;

        var bufRightLeft = node.Right.Left;

        var bufRightRight = node.Right.Right;

        node.Data = node.Right.Data;

        node.Right = bufRightRight;

        node.Left = bufRightLeft;

        Insert(bufLeft, node, node);

    }

    else

    {

        node.Right.Parent = node.Parent;

        Insert(node.Left, node.Right, node.Right);

    }

}