İSİM: NATHANAELLE BOPTI NGAH BONG
ÖĞRENCİ NUMERA: 24080410150
BILGISAYAR MÜHENDİSLİĞİ

İKİLİ AĞAÇ ÖDEVİ

```
public class TreeNode
{
    public int Data;
    public TreeNode left;
    public TreeNode right;

    public TreeNode(int data)
    {
        Data = data;
        left = null;
        right = null;
    }
}
public class BinaryTree
{
    public TreeNode root;

    public BinaryTree()
    {
        root = null;
    }
    public void Insert(int value)
    {
        TreeNode newNode = new TreeNode(value);
        if (root == null)
        {
            root = newNode;
            Console.WriteLine($"{value} Başarılı eklendi");
            return;
        }
        TreeNode current = root;
        while (current != null)
        {
            if (value < current.Data)
            {
                if (current.left == null)
                {
                    current.left = newNode;
                    Console.WriteLine($"{value} Başarılı eklendi");
                    break;
                }
                current = current.left;
            }
            else if (value > current.Data)
```

```
        {
            if (current.right == null)
            {
                current.right = newNode;
                Console.WriteLine($"{value} Başarılı eklendi");
                break;
            }
            current = current.right;
        }
        else
        {
            Console.WriteLine($"{value} zaten ağaçta");
            break;
        }
    }
}
public void yaz(string order)
{
    if (root == null)
    {
        Console.WriteLine("Agaç boştur");
        return;
    }
    switch (order)
    {
        case "Pre-order":
            preOrder(root); break;
        case "In-order":
            inOrder(root); break;
        case "Post-order":
            postOrder(root); break;
        case "Level-order":
            levelOrder(); break;
    }
    Console.WriteLine();
}
private void preOrder(TreeNode root)
{
    if (root != null)
    {
        Console.Write($"{root.Data} ");
        preOrder(root.left);
        preOrder(root.right);
    }
}
private void inOrder(TreeNode root)
{
    if (root != null)
    {
        inOrder(root.left);
        Console.Write($"{root.Data} ");
        inOrder(root.right);
    }
}
private void postOrder(TreeNode root)
{
    if (root != null)
    {
```

```
            postOrder(root.left);
            postOrder(root.right);
            Console.Write($"{root.Data} ");
        }
    }
    private void levelOrder()
    {
        var nodes = new List<TreeNode> { root };
        int i = 0;

        while (i < nodes.Count)
        {
            TreeNode current = nodes[i];
            Console.Write($"{current.Data} ");

            // Add children to the list
            if (current.left != null)
                nodes.Add(current.left);
            if (current.right != null)
                nodes.Add(current.right);

            i++;
        }
    }
}
public class Program
{
    static void Main()
    {
        BinaryTree tree = new BinaryTree();
        while (true)
        {
            Console.Write("Bir değer girin (ikili ağacı yazdırmak için 'yaz' girin)
: ");
            string input = Console.ReadLine();
            if (input == "yaz")
            {
                break;
            }
            else if (int.TryParse(input, out int value))
            {
                tree.Insert(value);
            }
            else
            {
                Console.WriteLine("Geçerli bir değer girin!");
            }

        }
        Console.WriteLine();

        Console.Write("Pre-order: ");
        tree.yaz("Pre-order");

        Console.Write("In-order: ");
        tree.yaz("In-order");

        Console.Write("Post-order: ");
```

```
        tree.yaz("Post-order");

        Console.Write("Level-order: ");
        tree.yaz("Level-order");

    }
}
```

```
Bir deger girin (ikili agaci yazdirmak için 'yaz' girin) : 1
1 Basarili eklendi
Bir deger girin (ikili agaci yazdirmak için 'yaz' girin) : 3
3 Basarili eklendi
Bir deger girin (ikili agaci yazdirmak için 'yaz' girin) : 4
4 Basarili eklendi
Bir deger girin (ikili agaci yazdirmak için 'yaz' girin) : 7
7 Basarili eklendi
Bir deger girin (ikili agaci yazdirmak için 'yaz' girin) : 3
3 zaten agaçta
Bir deger girin (ikili agaci yazdirmak için 'yaz' girin) : 9
9 Basarili eklendi
Bir deger girin (ikili agaci yazdirmak için 'yaz' girin) : 5
5 Basarili eklendi
Bir deger girin (ikili agaci yazdirmak için 'yaz' girin) : 2
2 Basarili eklendi
Bir deger girin (ikili agaci yazdirmak için 'yaz' girin) : yaz

Pre-order: 1 3 2 4 7 5 9
In-order: 1 2 3 4 5 7 9
Post-order: 2 5 9 7 4 3 1
Level-order: 1 3 2 4 7 5 9
```