```
using System;
using System.Collections;
using System.Collections.Generic;

namespace Practica30
{
    class Program
    {
        public class AvlArbol<TKey, TValue> : IEnumerable<TValue>
        {
            private IComparer<TKey> _comparer;
            private AvlNodo _root;

            sealed class AvlNodo
            {
                public AvlNodo Parent;
                public AvlNodo Left;
                public AvlNodo Right;
                public TKey Key;
                public TValue Value;
                public int Balance;
            }

            public AvlArbol(IComparer<TKey> compara)
            {
                _comparer = compara;
            }

            public AvlArbol() : this(Comparer<TKey>.Default)
            {
            }
            private AvlNodo RotateRight(AvlNodo node)
            {
                AvlNodo left = node.Left;
                AvlNodo leftRight = left.Right;
                AvlNodo parent = node.Parent;

                left.Parent = parent;
                left.Right = node;
                node.Left = leftRight;
                node.Parent = left;

                if (leftRight != null)
                {
                    leftRight.Parent = node;
                }
```

```
    if (node == _root)
    {
        _root = left;
    }
    else if (parent.Left == node)
    {
        parent.Left = left;
    }
    else
    {
        parent.Right = left;
    }

    left.Balance--;
    node.Balance = -left.Balance;

    return left;
}

private AvlNodo RotateLeftRight(AvlNodo node)
{
    AvlNodo left = node.Left;
    AvlNodo leftRight = left.Right;
    AvlNodo parent = node.Parent;
    AvlNodo leftRightRight = leftRight.Right;
    AvlNodo leftRightLeft = leftRight.Left;

    leftRight.Parent = parent;
    node.Left = leftRightRight;
    left.Right = leftRightLeft;
    leftRight.Left = left;
    leftRight.Right = node;
    left.Parent = leftRight;
    node.Parent = leftRight;

    if (leftRightRight != null)
    {
        leftRightRight.Parent = node;
    }

    if (leftRightLeft != null)
    {
        leftRightLeft.Parent = left;
    }
```

```
    if (node == _root)
    {
        _root = leftRight;
    }
    else if (parent.Left == node)
    {
        parent.Left = leftRight;
    }
    else
    {
        parent.Right = leftRight;
    }

    if (leftRight.Balance == -1)
    {
        node.Balance = 0;
        left.Balance = 1;
    }
    else if (leftRight.Balance == 0)
    {
        node.Balance = 0;
        left.Balance = 0;
    }
    else
    {
        node.Balance = -1;
        left.Balance = 0;
    }

    leftRight.Balance = 0;

    return leftRight;
}

private void InsertarBalance(AvlNodo node, int balance)
{
    while (node != null)
    {
        balance = (node.Balance += balance);

        if (balance == 0)
        {
            return;
        }
        else if (balance == 2)
        {
```

```
            if (node.Left.Balance == 1)
            {
                RotateRight(node);
            }
            else
            {
                RotateLeftRight(node);
            }

            return;
        }
    }
        AvlNodo parent = node.Parent;

        if (parent != null)
        {
            balance = parent.Left == node ? 1 : -1;
        }
        node = parent;
    }

public IEnumerator<TValue> GetEnumerator()
{
    throw new NotImplementedException();
}

IEnumerator IEnumerable.GetEnumerator()
{
    throw new NotImplementedException();
}
public void Insertar(TKey key, TValue value)
{
    if (_root == null)
    {
        _root = new AvlNodo { Key = key, Value = value };
    }
    else
    {
        AvlNodo node = _root;

        while (node != null)
        {
            int compare = _comparer.Compare(key, node.Key);

            if (compare < 0)
            {
```

```
                    AvlNodo left = node.Left;
                    if (left == null)
                    {
                        node.Left = new AvlNodo { Key = key, Value = value, Parent = node };
                        InsertarBalance(node, 1);
                        return;
                    }
                    else
                    {
                        node = left;
                    }
                }
                else if (compare > 0)
                {
                    AvlNodo right = node.Right;

                    if (right == null)
                    {
                        node.Right = new AvlNodo { Key = key, Value = value, Parent = node };

                        InsertarBalance(node, -1);

                        return;
                    }
                    else
                    {
                        node = right;
                    }
                }
                else
                {
                    node.Value = value;

                    return;
                }
            }
        }
    }

    static void Main(string[] args)
    {
    }
  }
}
```