Przechodzenie drzewa

Na zajęciach stworzyliśmy następującą klasę `Node<T>` reprezentującą węzeł drzewa:

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

namespace Generic {
    public class Node<T> {
        T data;
        IEnumerable<Node<T>> children;

        public T Data {
            get { return data; }
        }
        public IEnumerable<Node<T>> Children {
            get { return children; }
        }

        public Node(T data, IEnumerable<Node<T>> children) {
            this.data = data;
            this.children = children;
        }
        public Node(T data) : this(data, Enumerable.Empty<Node<T>>()) { }
    }
}
```

Proszę napisać funkcję, która dla drzewa o zadanym korzeniu generuje ciąg (zgodny z interfejsem `IEnumerable<T>`) wartości przechowywanych w węzłach tego drzewa w porządku **pre-order** (czyli najpierw wartość w korzeniu, a potem rekurencyjnie wartości w poszczególnych poddrzewach). Powinna to być publiczna statyczna metoda o sygnaturze `IEnumerable<T> Traverse<T>(Node<T> root)` znajdująca się w publicznej klasie `Tree` w przestrzeni nazw `Generic`. Jako rozwiązanie zadania proszę wysłać plik `.cs` zawierający kod tej metody. Oto przykładowy szablon takiego pliku:

```
using System.Collections.Generic;

namespace Generic {
    public class Tree {
        public static IEnumerable<T> Traverse<T>(Node<T> root) {
            // kod metody
        }
    }
}
```

Ciąg wynikowy powinien być generowany "leniwie" podczas przeglądania. W szczególności funkcja `Traverse` nie powinna sama z siebie powodować przeglądania całego drzewa.

Rozwiązanie będzie oceniane na dwóch testach: pierwszym poprawnościowym i drugim wydajnościowym. Zaliczenie tylko testu poprawnościowego pozwala na uzyskanie połowy "punktów" przewidzianych za to zadanie.