Examen Mundial de Programación Curso 2005-2006

Árboles de Prefijo. (Trie)

Un "Trie" (de re**trie**val) o "árbol de prefijos", es una estructura que se utiliza para almacenar palabras como cadenas de caracteres, de modo que a partir de una cadena, se puedan obtener de forma eficiente todas las palabras que tengan a dicha cadena como prefijo.

Un prefijo es cualquier porción del inicio de una palabra. (Note que toda palabra tiene a cadena vacía "" como prefijo y una palabra en sí es prefijo de ella misma).

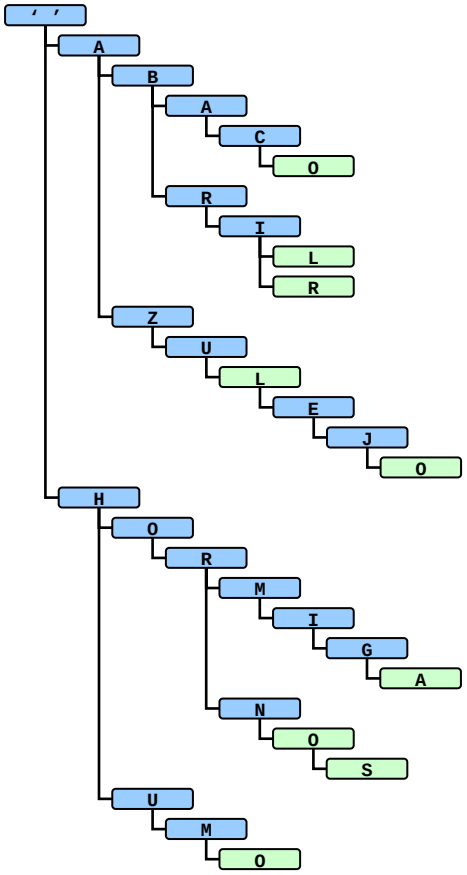

Figura 1 Un Trie que almacena palabras que comienzan con 'A' y con 'H'

En un Trie cada nodo tiene como valor un carácter (letra). En la figura 1 los nodos coloreados en verde indican que ahí hay un fin de palabra. Note que NO NECESARIAMENTE estos nodos son hojas, como es el caso del nodo que contiene a 'L', pues indica terminación de la palabra "AZUL" pero a su vez es parte de la palabra "AZULEJO". El `trie` de la figura 1 contiene entonces las palabras: "ABACO", "ABRIL", "ABRIR", "AZUL", "AZULEJO", "HORMIGA", "HORNO", "HORNOS", "HUMO". Las palabras que se forman a partir de un mismo nodo tienen un prefijo común. Ej: las palabras que tienen "AB" como prefijo son aquellas que se derivan del nodo que contiene el caracter 'B' que a su vez es hijo del nodo que contiene el carácter 'A', es decir: "**AB**ACO", "**AB**RIL", "**AB**RIR".

Defina una clase `Trie` que implemente la interface `ITrie` que se ofrece en la `dll` `Weboo.Utils.dll`

```
namespace Weboo.Utils {
      public interface ITrie {
            /// Adiciona una nueva palabra al Trie.
            /// Si la palabra es null el método lanza ArgumentNullException.
            /// Si la palabra ya existía el método no hace nada.
            void AgregaPalabra(string palabra);

            /// Cantidad de Palabras almacenadas en el Tire
            int CantidadDePalabras {get;}

            /// Enumerador de las palabras que tienen un prefijo dado.
            IEnumerator CadenasDePrefijo(string prefijo);

            /// Devuelve el TrieNode (hijo de la raiz del arbol) asociado
            /// al caracter dado. Si no se ha introducido ninguna palabra
            /// que comience por ese caracter, se retornará null.
            TrieNode this[char valorNodo]{get;}
      }
}
```

La clase `Trie` deberá quedar compilada en una `dll` y estar en el `namespace` `Weboo.ExamenMundial`.

Ejemplo. Si se hace:

```
ITrie t = new Trie ();
t.AgregaPalabra(“ABACO”);
t.AgregaPalabra(“ABRIL”);
t.AgregaPalabra(“ABRIR”);
t.AgregaPalabra(“HORMIGA”);
t.AgregaPalabra(“AZULEJO”);
t.AgregaPalabra(“AZUL”);
t.AgregaPalabra(“HUMO”);
t.AgregaPalabra(“HORNO”);
t.AgregaPalabra(“HORNOS”);
TrieNodo nodo1 = t[‘A’];
TrieNodo nodo2 = t[‘N’];
```

La variable `nodo1` referenciará al `TrieNode` con valor `‘A’` (ver Figura 1) mientras que `nodo2` tendrá valor `null` pues en el `trie t` no hay ninguna palabra que empiece con ‘N’, es decir no hay un nodo que contenga el carácter ‘N’ y que sea hijo del nodo raíz ‘’.

Si se hace ahora: `IEnumerator e = t.CadenasDePrefijo(“ABR”);`
la variable `e` referenciará a un enumerador que itera (en cualquier orden) sobre todas las cadenas incluidas en `t` que empiezan con el prefijo `ABR`, es decir, las cadenas “ABRIL” y “ABRIR”

La `dll` `Weboo.Utils.dll` ofrece la implementación de los tipos `TrieNode` y `TrieNodeList` que se muestran a continuación:

```
public class TrieNode {
    /// Devuelve el caracter del nodo
    public char Valor { get {…}}

    /// Indica si el nodo representa o no un fin de palabra
    public bool FinDePalabra { get {…} set {…} }

    /// Devuelve una lista con los TrieNode hijos del nodo.
    public TrieNodeList Hijos { get {…} }

    /// Crea un nodo con el valor especificado
    public TrieNode(char value,bool esFinPalabra,params TrieNode[] hijos){…}
  }
```

Donde `TrieNodeList` es:

```
public class TrieNodeList : IEnumerable {
    /// Adiciona un TrieNode a la lista
    /// Lanza ArgumentNullException si node es null.
    public void Add(TrieNode node);

    /// Cantidad de TrieNodes en la lista
    public int Count { get {…} }

    /// Devuelve el TrieNode que se encuentra en la posicion index.
    /// Lanza IndexOutOfRangeException si index<0 o index>=Count.
    public TrieNode this[int index] { get {…} }

    /// Devuelve el TrieNode cuyo valor es c.
    /// null si el nodo no existe.
    public TrieNode this[char c] { get {…} }
}
```