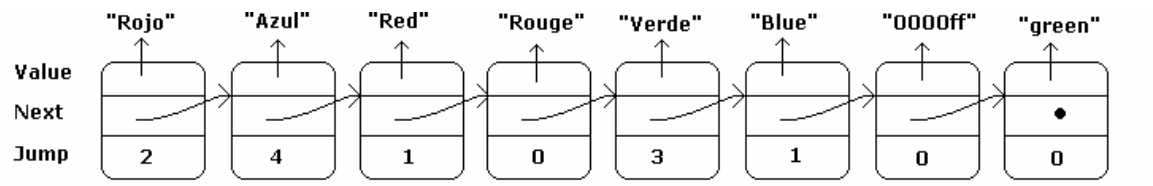
Universidad de la Habana.
Facultad de Matemática y Computación.
Carrera Ciencia de la Computación, Asignatura Programación
3er Problema de Programación, Curso 2006 - 2007

Salta, salta, salta

Una manera de representar varias listas (donde cada una representa una categoría o grupo de objetos) usando una única estructura, es la siguiente:



En esta estructura en forma de lista, cada nodo tiene tres campos: una referencia, `Next`, al próximo nodo, una referencia, `Value`, a un objeto que se considera el valor del nodo (en la Figura de arriba estos valores son cadenas pero note que pueden ser objetos de cualquier tipo) y un tercer campo de tipo entero, `Jump`, cuyo valor (que se asegura es no negativo) indica cuántos nodos más adelante se encuentra el nodo cuyo valor pertenece a la misma categoría de objetos. Un mismo nodo no puede pertenecer a dos categorías diferentes.
Por ejemplo, el la figura anterior, el "Rojo" es el primer elemento de su categoría, y el salto de 2 indica que dos nodos más adelante aparece el "Red", y el salto de 1 en éste indica al ¨Rouge¨ como el siguiente y último, por tener su salto con valor 0. Los nodos que no aparezcan en la categoría de los rojos, pertenecen a sus propias categorías, formando nuevas listas de elementos.

Estos nodos están representados por la clase `Node` de la biblioteca `Weboo.Structures` que se brinda junto a este documento. La clase `Node` se muestra a continuación:

```
public class Node
 {
     //devuelve el valor asociado al nodo
     public object Value;
     //devuelve el salto asociado al nodo
     public uint Jump;
     //devuelve una referencia al próximo nodo
     public Node Next;
     ...
 }
```

El tipo uint es muy parecido a int, pero solo permite valores positivos. Utilice variables de tipo uint para guardar este tipo de valores. Por ejemplo:
uint size = node.Jump;
for(uint i = 0; i < node.Jump; i++) {…}
etc…

Programe una clase `CategoryEnumerator` en el espacio de nombres `Pregunta3` que implemente la interfaz `ICategoryEnumerator`, también brindada en la biblioteca y cuya definición se muestra a continuación. Note que la interfaz `ICategoryEnumerator` hereda de `IEnumerator`, por lo tanto, su clase será un enumerador con una operación adicional. La clase deberá ser implementada dentro de un proyecto de tipo "*Class Library*" con nombre Pregunta3. (y no **Pregunta3.dll**)

```
namespace Pregunta3
{
       public interface ICategoryEnumerator : Ienumerator
       {
           bool MoveNextCategory();
       }
}
```

Adicionalmente a las operaciones habituales aplicables a todo enumerador, un `ICategoryEnumerator` permitirá avanzar hasta el primer objeto de la siguiente categoría mediante el método `MoveNextCategory`.

El constructor de su clase debe tener la siguiente signatura:

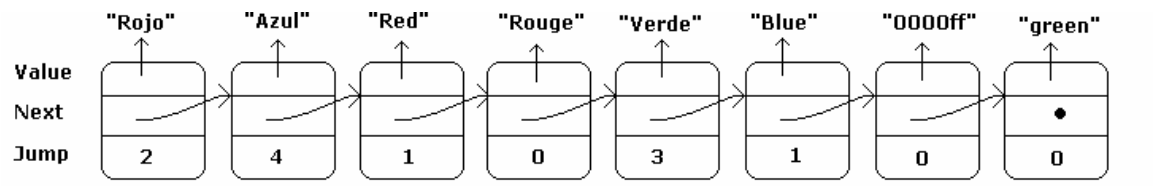
```
public CategoryEnumerator (Node first)
```

Como se observa, una instancia de `CategoryEnumerator` se construirá a partir de un `Node`, que representará al primer nodo de la estructura.
El propósito del `CategoryEnumerator` es iterar sobre los objetos que se encuentren en la lista original, pero **organizados por categorías**, o sea, los objetos correspondientes a la primera categoría serán devueltos antes de los correspondientes a la segunda categoría y así sucesivamente. La precedencia entre categorías estará determinada por el orden en que aparezca el primer objeto de cada categoría en la lista de nodos original.

La enumeración implementada en `CategoryEnumerator` debe ser la siguiente:

- El primer valor devuelto es el valor del primer nodo de la lista, que corresponde con el primer valor de la primera categoría.
- Si en el paso i de la enumeración se devolvió el valor de un nodo N, entonces en el paso i + 1 se devolverá el valor de aquél nodo que represente el próximo que pertenezca a la misma categoría que N (note que para ello se dispone del campo `Jump`).
- Si en el paso i de la enumeración se devolvió el valor de un nodo N y éste tiene 0 como valor del campo `Jump`, entonces el valor a devolver en el paso i + 1 es el del primer nodo de la próxima categoría.
- La enumeración debe terminar cuando se hayan devuelto todos los valores de todas las categorías.
- Si durante la enumeración de una categoría se utiliza el método `MoveNextCategory`, el enumerador debe saltar el resto de los valores de la categoría actual y posicionar al `Current` en el primer elemento de la siguiente categoría, si es que hay una siguiente, o debe terminar la iteración si era la última. Si estaba posicionado en el último nodo de la categoría no salta ningún nodo y simplemente pasa al primer nodo de la siguiente categoría.

Por ejemplo, para la estructura siguiente



la enumeración debe ser

`“Rojo”, “Red”, “Rouge”, “Azul”, “Blue”, “0000ff”, “Verde”, “green”.`

Aclaraciones:

- Dos objetos estarán en la misma categoría si se puede llegar desde uno de ellos hasta el otro caminando desde el nodo donde está el primero hacia el nodo donde está el segundo usando el valor del campo `Jump`.
- Antes de haber ejecutado `MoveNext`, aplicar `MoveNextCategory` a un `ICategoryEnumerator` provoca que se avance hasta el primer objeto de la primera clasificación (y pasa a ser legal aplicar la propiedad `Current`).