

Programación Declarativa 2007-2008 Práctica 8

Si se considera que **Empty** representa el árbol vacío y que **Node (r,i,d)** representa el árbol que tiene raíz **r**, rama izquierda **i** y rama derecha **d**, el tipo de dato **'a bintree**, definido a continuación, serviría para representar árboles binarios cuyos nodos están etiquetados con valores de tipo 'a.

```
type 'a bintree = Empty | Node of 'a * 'a bintree * 'a bintree;;
```

Defina en ocaml las siguientes funciones:

```
raiz: 'a bintree -> 'a
que devuelva el valor de la raíz del árbol
```

```
ramaizda: 'a bintree -> 'a bintree
la rama izquierda del árbol
```

```
ramadcha: 'a bintree -> 'a bintree
la rama derecha del árbol
```

```
altura: 'a bintree -> int
el número de niveles del árbol(atención: altura del árbol vacío = 0)
```

```
num_nodos: 'a bintree -> int
el número de nodos
```

```
t_max: 'a bintree -> 'a
el valor máximo de los nodos
```

```
t_min: 'a bintree -> 'a
el valor mínimo de los nodos
```

```
t_sum: int bintree -> int
la suma de todos los nodos
```

```
t_spec: 'a bintree -> 'a bintree
la imagen especular del árbol
```

```
hojas: 'a bintree -> 'a list
la lista de hojas del árbol, de izda a derecha
```

```
preorden: 'a bintree -> 'a list
todos los nodos del árbol en pre-orden
```

```
postorden: 'a bintree -> 'a list
todos los nodos del árbol en post-orden
```

```
inorden: 'a bintree -> 'a list
todos los nodos del árbol en in-orden
```

```
anchura: 'a bintree -> 'a list
todos los nodos del árbol según un recorrido "en anchura" de arriba a abajo y de izquierda a derecha.
```

