Facultad de Ciencias Exactas, Ingeniería y Agrimensura
Escuela de Ciencias Exactas y Naturales
Departamento de Ciencia de la Computación
Estructura de Datos y Algoritmos II

Nombre y Apellido: Legajo:

Examen Final

1. Se representan secuencias mediante árboles binarios dados por el siguiente tipo de datos:

data BTree a = E | L a | N *Int* (BTree a) (BTree a)

donde el recorrido *inorder* del árbol da el orden de los elementos de la secuencia y el valor entero guardado en los nodos representa la longitud de la misma.
Definir en Haskell, de manera eficiente, la función stripSufix :: *Eq* $a \Rightarrow$ BTree $a \to$ BTree $a \to$ Maybe (BTree a), que elimine el sufijo dado en una secuencia, devolviendo Nothing si la secuencia no termina con el sufijo dado o Just xs donde xs es la secuencia que queda antes del sufijo.
Por ejemplo,

stripSufix$\langle$'b', 'a', 'r'$\rangle\langle$'f', 'o', 'o', 'b', 'a', 'r'$\rangle$ = Just$\langle$'f', 'o', 'o'$\rangle$
stripSufix$\langle$'b', 'a', 'r'$\rangle\langle$'f', 'o', 'o', 'b', 'a', 'r', 'r'$\rangle$ = Nothing

2. Cierto periódico quiere manipular los resultados de las elecciones de las paso en Argentina asociando los resultados por ciudades cercanas de manera conveniente a un partido X.
Para ello, dada una secuencia s con los balances de votos de un partido X, se desea encontrar la suma de una subsecuencia de s que maximice las sumas de dichos balances. Definir una función maxBalance : *Seq Int* $\to$ *Int* que resuelva el problema.
Por ejemplo,

maxBalance$\langle -2, -4, 6, -1, 5, -7, 2\rangle = 10$

Definir maxBalance usando funciones del Tad secuencias, con profundidad de orden $\lg n$.

3. Dado el TAD *Array*, con las siguientes operaciones:

tad *Array* (V : *Set*) **where**
import *Int*, *Bool*

new	: *Int* $\to$ V $\to$ *Array* V	-- dados n y v crea un arreglo de dimension n e inicializa -- cada elemento con v
set	: *Int* $\to$ V $\to$ *Array* V $\to$ *Array* V	-- modifica el valor asociado a un índice del arreglo
dimension	: *Array* V $\to$ *Int*	
get	: *Int* $\to$ *Array* V $\to$ V	-- obtiene el valor almecenado en un índice del arreglo
slicing	: *Int* $\to$ *Int* $\to$ *Array* V $\to$ *Array* V	-- dados i y j enteros y un arreglo a, crea un arreglo con -- valores a[i] .. a[j]

Dar una especificación algebraica para el TAD.

4. (Lólo libres)Dadas las siguientes definiciones:

$$\textbf{data}\ \mathsf{Rose}\ a = \mathsf{E} \mid \mathsf{N}\ a\ [\mathsf{Rose}\ a]$$
$$\mathsf{flatten} :: \mathsf{Rose}\ a \to [a]$$
$$\mathsf{flatten}\ \mathsf{E} = [\,]$$
$$\mathsf{flatten}\ (\mathsf{N}\ x\ xs) = x : \mathsf{concat}\ (\mathsf{map}\ \mathsf{flatten}\ xs)$$
$$\mathsf{mapRose} :: (a \to b) \to \mathsf{Rose}\ a \to \mathsf{Rose}\ b$$
$$\mathsf{mapRose}\ f\ \mathsf{E} = \mathsf{E}$$
$$\mathsf{mapRose}\ f\ (\mathsf{N}\ x\ xs) = \mathsf{N}\ (f\ x)\ (\mathsf{map}\ (\mathsf{mapRose}\ f)\ xs)$$

probar por inducción estructural que $\mathsf{map}\ f \circ \mathsf{flatten} = \mathsf{flatten} \circ \mathsf{mapRose}\ f$
Ayuda: Puede utilizar el siguiente lema: $\mathsf{concat} \circ \mathsf{map}\ (\mathsf{map}\ f) = \mathsf{map}\ f \circ \mathsf{concat}$