UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

Estructuras Discretas

Tarea 5

PRESENTA

Castañon Maldonado Carlos Emilio
Bazán Rojas Karina Ivonne

PROFESORA

Araceli Liliana Reyes Cabello

AYUDANTES

Rafael Reyes Sánchez
Ricardo Rubén Gónzalez García
José Eliseo Ortíz Montaño
Javier Enríquez Mendoza

Estructuras Discretas

Tarea Semanal 5

1 Mediante inducción matemática demuestre que el número de nodos de un árbol binario en el nivel i-ésimo es a lo más 2^{i-1}

Paso Base:
Consideremos como nuestro paso base a un árbol de un solo nodo, es decir a un árbol al que solo lo conforme la raíz, por lo que tendríamos a $i = 1$
$2^{1-1} = 2^0 = 1$
Hipótesis Inductiva: Suponemos que se cumple para i, es decir:
2^{i-1}
Paso Inductivo: Probaremos que se cumple para el siguiente elemento $i + 1$, es decir:
$2^{(i+1)-1}$
Primero que nada notemos que $2^{(i+1)-1} = 2^{i+1-1} = 2^i$, además de que podemos observar que por definicion, cada nodo en el nivel i tiene a lo mas dos nodos hijos en el nivel $i + 1$, por ende el número total a lo mas de nodos en el nivel $i + 1$ es igual a lo mas 2 veces el número de nodos en el nivel i.
Ahora, usando nuestra hipótesis de inducción sabemos que el número de nodos en el nivel i es a lo más 2^{i-1} por lo tanto, el número de nodos en el nivel $i + 1$ es a lo más $2 \cdot 2^{i-1} = 2^i$.
Por lo tanto, queda demostrado por el principio de inducción que: el nivel i-ésimo de un árbol binario completo tiene a lo más 2^{i-1} nodos.

2 Realiza una función recursiva llamada nodosNivel, la cual recibe un árbol binario y un número que representa el nivel y devuelve una lista con todos los nodos del árbol que se encuentran en ese nivel.
Si consideramos el árbol binario t1, véase la figura 1

$$t1 \;=\; AB\ 1\ (AB\ 2\ (AB\ 4\ (H\ 8)\ (H\ 9))\ (H\ 5))\ (AB\ 3\ (AB\ 6\ (H\ 10)\ (H\ 11))\ (AB\ 7\ Vacio\ (H\ 12)))$$

Ejemplo: **nodosNivel** $2\ t1 \;=\; [4, 5, 6, 7]$

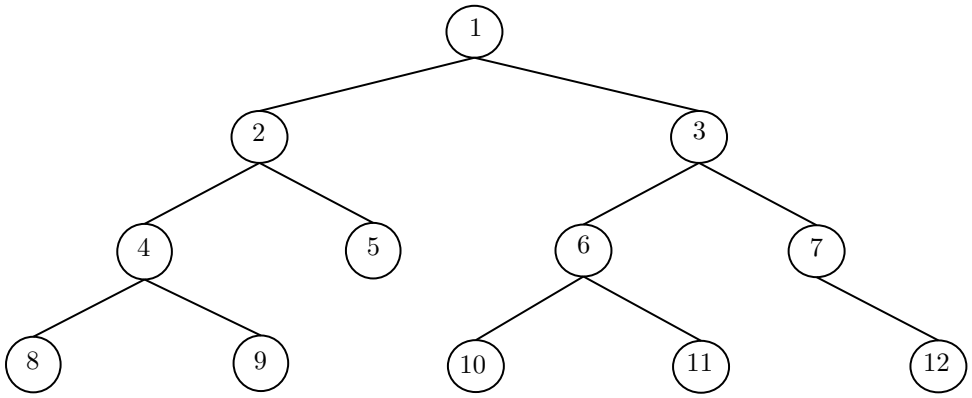

Figura 1: Árbol binario

```
data ArbolB = Vacío | H Int | AB Int ArbolB ArbolB deriving (Show)
nodosNivel :: ArbolB -> Int -> [Int]
nodosNivel Vacío _ = []
nodosNivel (H a) 1 = [a]
```

```
nodosNivel (AB r izq der) k
    | k <= 0 = []
    | k == 1 = [r]
    | otherwise = nodosNivel izq (k - 1) ++ nodosNivel der (k - 1)
```

3 El recorrido de un árbol es el proceso que permite acceder una sola vez a cada uno de sus nodos. Existen diferentes formas de hacer el recorrido de un árbol binario, ya sea en anchura o en profundidad.
Recorrido en Profundidad
PreOrden:

a) Primero se visita su raíz.

b) Después el subárbol izquierdo en preOrden.

c) Al final el subárbol derecho en preOrden.

Por ejemplo, si consideramos el árbol de la figura 1 el recorrido en preOrden sería:

$$1, 2, 4, 8, 9, 5, 3, 6, 10, 11, 7, 12$$

De acuerdo con la definición de recorrido en preOrden, genera una función recursiva que imprime el recorrido en preOrden en una lista de un árbol binario.
Por ejemplo: **preOrdenAB** $t1 = [1, 2, 4, 8, 9, 5, 3, 6, 10, 11, 7, 12]$

```
data ArbolB = Vacío | H Int | AB Int ArbolB ArbolB deriving (Show)
preOrdenAB :: ArbolB -> [Int]
preOrdenAB Vacío = []
preOrdenAB (H a) = [a]
preOrdenAB (AB r izq der) = [r] ++ preOrdenAB izq ++ preOrdenAB der
```