UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

Estructuras Discretas

Tarea 4

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 4

1 Por medio de inducción estructural demuestre que se cumple la siguiente propiedad en listas

$reversa\ (xs + +ys)\ =\ (reversa\ ys)\ + +\ (reversa\ xs)$

Inducción sobre xs
Caso Base:

$$\begin{aligned} xs &= [\,] \\ reversa\ ([\,]\ + +\ ys) &= reversa\ ys \quad \text{Por definición de concatenación} \\ &= reversa\ ys\ + +\ [\,] \quad \text{Por definición de concatenación y reversa} \end{aligned}$$

Hipótesis Inductiva: Suponiendo que se cumple para $xs = zs$
$reversa\ (zs\ + +\ ys)\ =\ reversa\ ys\ + +\ zs$
Paso Inductivo: Demostraremos que se cumple para $xs = (a : zs)$

$$\begin{aligned} reversa\ ((a : zs)\ + +\ ys) &= reversa\ (a : (zs\ + +\ ys)) \quad \text{Por definición de concatenación} \\ &= (reversa(zs\ + +\ ys))\ + +\ a \quad \text{Por definición de concatenación} \\ &= ((reversa\ ys)\ + +\ (reversa\ zs))\ + +\ [a] \quad \text{Por } \textbf{H.I} \\ &= reversa\ ys\ + +\ reversa\ (a : zs) \quad \text{Por definición de concatenación} \end{aligned}$$

$\therefore$ Por el principio de inducción estructural queda demostrada la igualdad:
$reversa\ (xs + +ys)\ =\ (reversa\ ys)\ + +\ (reversa\ xs)$

2 Genera una función recursiva de nombre *ocurrencias*, que recibe un carácter y una lista de carácteres y regresa el número de ocurrencias de éste carácter en la lista.
Por ejemplo: ocurrencias a [a,n,i,t,a,l,a,v,a,l,a,t,i,n,a] = 6
ocurrencias :: Char $\rightarrow [a] \rightarrow$ Int
Caso base:
ocurrencias $xs\ =\ xs$
ocurrencias $c\ [\,]\ =\ [\,]$
Caso recursivo:
ocurrencias $n\ (x : xs)\ =\ length\ aux\ (x : xs)$

aux :: Int $\rightarrow$ [String] $\rightarrow$ Int
$aux\ n\ (x : xs)\ =\ if\ n\ ==\ (head\ xs)\ then\ n\ :\ (head\ xs)\ else\ aux\ n\ (tail\ xs)$

3 Realiza una función recursiva de nombre *quitaElem*, que recibe un carácter y una lista de carácteres y regresa la lista sin que aparezca el carácter dado.
Por ejemplo: quitaElem a [a,n,i,t,a,l,a,v,a,l,a,t,i,n,a] = [n,i,t,l,v,l,t,i,n]
quitaElem :: Char $\rightarrow [a] \rightarrow [a]$
Caso base:
quitaElem $xs \ = \ xs$
quitaElem $c \ [\,] \ = \ [\,]$
Caso recursivo:
quitaElem $c \ (x : xs) \ = \ if \ c \ == \ (head \ xs) \ then \ quita` \ c \ else \ c \ (tail \ xs)$

quita` :: Char $\rightarrow [a] \rightarrow [a]$
$quita` \ [\,] \ = \ [\,]$
$quita` \ (x \ : \ xs) \ = \ if \ (xs == [\,]) \ then \ x \ else \ (quita` xs)$