Ignacio Sebastián Molina

Ejercicio 1 (Apartado 5 según hoja de examen)

```
infoCo :: (Seq s) => S (Int, Int) -> (Int, S Int)
infoCo s = let  seq' = mapS (\(x,y) -> x-y) s
                seq'' = tabulateS (\i -> (fst (nthS s i), i+1)) (lengthS s)
                (registro, red) = scanS (+) 0 seq'
                regFinal = appendS (dropS registro 1) (singletonS red)
                mayorTupla = reduceS (\(x,y) (x',y') -> if x > x'
                                                        then (x,y)
                                                        else (x',y'))
                                     (0,0) seq''
           in (snd (mayorTupla), regFinal
```

Ejercicio 2 (Apartado 3 según hoja de examen)

```
data Tree a = E | L a | N (Tree a) (Tree a)

partir :: Tree a -> Tree (Tree a, a, Tree a)

partir arbol = aux arbol E E  where

    aux E _ _ = E

    aux (L dato) izqAcum derAcum = L (izqAcum, dato, derAcum)

    aux (N izq der) izqAcum derAcum =
        let (l, r) = aux izq izqAcum (N der derAcum) |||
                     aux der (N izqAcum izq) derAcum
        in N (l, r)
```