```haskell
module LKSA_2016_01_11 where
import Data.List

-------------------------------------
-- MURGILKETA
-------------------------------------

nartzisista_lag :: Integer -> Integer -> Integer -> Integer -> Bool
nartzisista_lag x n a s
    | (x < (10^(n-1))) || (x >= (10^n)) = error "Digitu-kopurua ez da zuzena"
    | (a == 0) && (x==s) = True
    | (a == 0) = False
    | otherwise = nartzisista_lag x n (div a 10) (s+((mod a 10)^3))

-------------------------------------

nartzisista :: Integer -> Integer -> Bool
nartzisista x n
    | (x < (10^(n-1))) || (x >= (10^n)) = error "Digitu-kopurua ez da zuzena"
    | otherwise = nartzisista_lag x n x 0

-------------------------------------



-------------------------------------
-- BUKAERAKO ERREKURTSIBITATEA
-------------------------------------

atzeratu :: Integer -> [Integer] -> [Integer]
atzeratu x s
    | (null s) = []
    | (x == (head s)) = (atzeratu x (tail s)) ++ [x]
    | otherwise = (head s):(atzeratu x (tail s))

-------------------------------------

atzeratu_lag :: Integer -> [Integer] -> [Integer] -> [Integer] -> [Integer]
atzeratu_lag x s p q
    | (null s) = p++q
    | (x == (head s)) = atzeratu_lag x (tail s) p (x:q)
    | otherwise = atzeratu_lag x (tail s) (p++[(head s)]) q

atzeratu_be :: Integer -> [Integer] -> [Integer]
atzeratu_be x s = atzeratu_lag x s [] []

-------------------------------------
```

```
-------------------------------------
-- ZERRENDA-ERAKETA
-------------------------------------

luzerak :: [[Integer]] -> [Integer]
luzerak s = [ (genericLength x) | x <- s ]

-------------------------------------

luzeena :: [[Integer]] -> [[Integer]]
luzeena s = [ x | x <- s, (genericLength x) == (maximum (luzerak s)) ]

-------------------------------------

sekuentzia :: [Integer] -> Integer -> Integer -> [Integer]
sekuentzia s i n
    | (i < 1) || ((i+n-1) > (genericLength s)) = error "Indizea edo luzera ez dira zuzenak"
    | otherwise = (genericTake n (genericDrop (i-1) s))

-------------------------------------

azpisekuentziak :: [Integer] -> [[Integer]]
azpisekuentziak s = [ (sekuentzia s i n) | n <- [1..(genericLength s)], i <- [1..((genericLength s)+1-n)] ]

-------------------------------------

aurrizkia :: [Integer] -> [Integer] -> Bool
aurrizkia r s = r == (genericTake (length r) s)

-------------------------------------

azpizerrenda :: [Integer] -> [Integer] -> Bool
azpizerrenda r s = or [ (aurrizkia r (genericDrop i s)) | i <- [0..((length s)-(length r)+1) ]]

-------------------------------------

bien_azpisekuentziak :: [Integer] -> [Integer] -> [[Integer]]
bien_azpisekuentziak r s = [ x | x <- (azpisekuentziak r), (azpizerrenda x s) ]

-------------------------------------
bien_azpisekuentzia_luzeenak :: [Integer] -> [Integer] -> [[Integer]]
bien_azpisekuentzia_luzeenak r s = luzeena (bien_azpisekuentziak r s)

-------------------------------------
```