```haskell
module Clase3
where

-- Pares
--------

pares :: [a] -> [(a,a)]
pares [] = []
pares (x:xs) = map ((,) x) xs ++ pares xs


-- Tramos
---------

prefijos :: [a] -> [[a]]
prefijos [] = [[]]
prefijos (x:xs) = []:map (x:) (prefijos xs)

tramos :: [a] -> [[a]]
tramos [] = [[]]
tramos (x:xs) = map (x:) (prefijos xs) ++ tramos xs

-- Listas infinitas

losNaturales :: Num a => [a]
losNaturales = iterate (+1) 0

primerosNaturales :: Num a => Int -> [a]
primerosNaturales n = take n losNaturales

multiplosDe :: Num a => a -> [a]
multiplosDe x = iterate (+x) 0

primerosMultiplos :: Num a => Int -> a -> [a]
primerosMultiplos n x = take n (multiplosDe x)
```