```haskell
-- | Some useful funcions.
module Utils (
    if', atLeast, flipL, flipR, select, combinations, merge
) where

import           Data.List (subsequences, transpose)

-- | If-Then-Else without syntactic sugar.
if' :: Bool -> a -> a -> a
if' True = const
if' _    = flip const

-- | Case without syntactic sugar.
select :: a -> [(Bool, a)] -> a
select = foldr $ uncurry if'

-- | Tests if given list's size is at least of n.
atLeast :: Int -> [a] -> Bool
atLeast 0 = const True
atLeast n = not . null . drop (n - 1)

-- | Flip for functions with 3 parameters. Puts the first one on last.
flipL :: (a -> b -> c -> d) -> b -> c -> a -> d
flipL f b c a = f a b c

-- | Flip for functions with 3 parameters. Puts the last one on first.
flipR :: (a -> b -> c -> d) -> c -> a -> b -> d
flipR f c a b = f a b c

-- | Generates the combinations of K distinct objects chosen from the N elements of a list.
combinations :: Int -> [a] -> [[a]]
combinations k = filter ((k==) . length) . subsequences

-- | Merges two lists.
merge :: [a] -> [a] -> [a]
merge l r = concat $ transpose [l, r]

```