Module Title: INFORMATICS 1 - FUNCTIONAL PROGRAMMING
Exam Diet (Dec/April/Aug): AUGUST 2009
Brief notes on answers:

import Char
import List
import Test.QuickCheck

f :: String -> Bool
f xs = and [ even (digitToInt x) | x <- xs, isDigit x ]

g :: String -> Bool
g [] = True
g (x:xs) | isDigit x = even (digitToInt x) && g xs
| otherwise = g xs

h :: String -> Bool
h xs = foldr (&&) True (map even (map digitToInt (filter isDigit xs)))

prop_f =
f "246" == True &&
f "2467" == False &&
f "x4y2z" == True &&
f "abc12" == False

prop_fg xs = f xs == g xs

prop_gh xs = g xs == h xs

ok_1 =
  quickCheck prop_f >>
  quickCheck prop_fg >>
  quickCheck prop_gh

p :: Int -> Int -> Int
p x y | x < y = length [ i | i <- [x+1 .. y-1], even i ]
      | x > y = length [ i | i <- [y+1 .. x-1], even i ]

r :: [Int] -> Int
r (x:y:xs) = maximum [ p u v | (u,v) <- zip (x:y:xs) (y:xs) ]

s :: [Int] -> Int
s [x,y] = p x y
s (x:y:z:xs) = p x y 'max' s (y:z:xs)
prop_p =
  p 3 7 == 2 &&
  p 7 8 == 0 &&
  p 8 0 == 3

prop_r =
  r [3,7,8,0] == 3

prop_rs xs = (length xs > 1 && distinct xs) ==> r xs == s xs
  where
  distinct xs = (nub xs == xs)

ok_2 =
  quickCheck prop_p >>
  quickCheck prop_r >>
  quickCheck prop_rs

prop_t =
  t "1234" == 4321 &&
  t "526" == 625

prop_tu xs = all isDigit xs ==> t xs == u xs

ok_3 =
  quickCheck prop_t >>
  quickCheck prop_tu