import Char
import Test.QuickCheck

-- Question 1

isPunctuation x = not (isAlpha x || isDigit x)

-- 1a

f :: String -> Bool
f xs = and [ x == ' ' | x <- xs, isPunctuation x ]

f2 :: String -> Bool
f2 xs = and [ isAlpha x || isDigit x || x == ' ' | x <- xs ]

-- 1b

g :: String -> Bool
g [] = True

g (x:xs) | isPunctuation x = x == ' ' && g xs
| otherwise = g xs

-- 1c

h :: String -> Bool
h = foldr (&&) True . map (== ' ') . filter isPunctuation

test1 =
  f "Just two spaces"    == True    &&
  f "No other punctuation, period." == False &&
  f "No exclamations!"    == False    &&
prop_1 :: String -> Bool
prop_1 xs = f xs == f2 xs && f2 xs == g xs && g xs == h xs

-- Question 2

-- 2a

p :: a -> [a] -> [a]
p x ys = x : concat [ [y,x] | y <- ys ]

p2 :: a -> [a] -> [a]
p2 x ys = concat [ [x,y] | y <- ys ] ++ [x]

-- 2b

q :: a -> [a] -> [a]
q x [] = [x]
q x (y:ys) = x : y : q x ys

test2 =
    p 'x' "ABCD" == "xAxBxCxDx" &&
p 'a' "XY" == "aAxAa" &&
p 'l' "Hello" == "-H-e-l-l-o-" &&
p '"' "" == "-" &&
p 0 [1,2,3,4,5] == [0,1,0,2,0,3,0,4,0,5,0]

prop_2 :: Int -> [Int] -> Bool
prop_2 x ys = p x ys == p2 x ys && p x ys == q x ys

-- Question 3
type Point = (Int,Int)
data Points = X
  | Y
  | DX Int Points
  | DY Int Points
  | U Points Points

-- 3a

inPoints :: Point -> Points -> Bool
inPoints (x,y) X = y == 0
inPoints (x,y) Y = x == 0
inPoints (x,y) (DX dx p) = inPoints (x-dx,y) p
inPoints (x,y) (DY dy p) = inPoints (x,y-dy) p
inPoints (x,y) (U p q) = inPoints (x,y) p || inPoints (x,y) q

test3a =
inPoints (3,0) X == True &&
inPoints (0,1) Y == True &&
inPoints (3,3) (DY 3 X) == True &&
inPoints (2,1) (DX 2 Y) == True &&
inPoints (3,0) (U X Y) == True &&
inPoints (0,1) (U X Y) == True &&
inPoints (3,3) (U (DY 3 X) (DX 2 Y)) == True &&
inPoints (2,1) (U (DY 3 X) (DX 2 Y)) == True &&
inPoints (3,0) (U (U X Y) (U (DX 2 Y) (DY 3 X))) == True &&
inPoints (0,1) (U (U X Y) (U (DX 2 Y) (DY 3 X))) == True &&
inPoints (3,3) (U (U X Y) (U (DX 2 Y) (DY 3 X))) == True &&
inPoints (2,1) (U (U X Y) (U (DX 2 Y) (DY 3 X))) == True &&
inPoints (1,1) X == False &&
inPoints (1,1) Y == False &&
inPoints (1,1) (DY 3 X) == False &&
inPoints (1,1) (DX 2 Y) == False &&
inPoints (1,1) (U X Y) == False &&
inPoints (1,1) (U X Y) == False &&
inPoints (1,1) (U (DY 3 X) (DX 2 Y)) == False &&
inPoints (1,1) (U (DY 3 X) (DX 2 Y)) == False &&
inPoints (1,1) (U (U X Y) (U (DX 2 Y) (DY 3 X))) == False
-- 3b

countAxes :: Points -> Int
countAxes X = 1
countAxes Y = 1
countAxes (DX dx p) = countAxes p
countAxes (DY dy p) = countAxes p
countAxes (U p q) = countAxes p + countAxes q

test3b =
  countAxes X == 1 &&
  countAxes Y == 1 &&
  countAxes (U X Y) == 2 &&
  countAxes (U (DY 3 X) (DX 2 Y)) == 2 &&
  countAxes (U (U X Y) (U (DX 2 Y) (DY 3 X))) == 4 &&
  countAxes (U (U X Y) X) == 3

-- Tests

testAll =
  test1 && test2 && test3a && test3b

checkAll =
  quickCheck prop_1 >>
  quickCheck prop_2