isSmallDigit x = digitToInt x >= 5

f :: String -> Bool
f xs = and [ isSmallDigit x | x <- xs, isDigit x]

-- 1b

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

-- 1c

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

-- Question 2

-- 2a

p :: [a] -> [a]
p [] = []
p xs = concat
    [ [y,x] | ((x,y),i) <- zip (zip xs (tail xs)) [1..], odd i ]

-- 2b

q :: [a] -> [a]
q [] = []
q (x:y:xs) = y : x : q xs
-- Question 3

type Point = (Int,Int)
data Points = Lines Int Int  
| Columns Int Int  
| Union Points Points  
| Intersection Points Points  

-- 3a

inPoints :: Point -> Points -> Bool  
inPoints (_,y) (Lines min max) = min <= y && y <= max  
inPoints (x,_) (Columns min max) = min <= x && x <= max  
inPoints p (Union ps qs) = inPoints p ps || inPoints p qs  
inPoints p (Intersection ps qs) = inPoints p ps && inPoints p qs  

-- 3b

showPoints :: Point -> Points -> [String]  
showPoints (a,b) ps = [ makeline y | y <- [0..b] ]  
    where  
        makeline y = [ if inPoints (x,y) ps then '*' else ' ' | x <- [0..a] ]