Haskell Assignment By: Justin Cesarini

Abstract

This project taught me a lot about haskell. Going in I knew little to nothing about Haskell but after this I feel comfortable programming in Haskell and I enjoyed it very much

Task 1

```
C:\DEV>ghci
GHCi, version 9.2.1: https://www.haskell.org/ghc/  :? for help
ghci>
ghci> length[2,3,5,7]
4
ghci> words "need more coffee"
["need","more","coffee"]
ghci> reverse "need more coffee"
"eeffoc erom deen"
ghci> reverse ["need", "more", "coffee"]
["coffee","more","need"]
ghci> head ["need", "more", "coffee"]
"need"
ghci> tail ["need", "more", "coffee"]
["more","coffee"]
ghci> last ["need", "more", "coffee"]
"coffee"
ghci> init ["need", "more", "coffee"]
["need","more"]
ghci> take 7 ["need", "more", "coffee"]
["need","more","coffee"]
ghci> take 7 "need", "more", "coffee"

<interactive>:11:14: error: parse error on input `,'
ghci> take 7 "need more coffee"
"need mo"
ghci> drop 7 "need more coffee"
"re coffee"
ghci> (\x -> length x > 5) "Friday"
True
ghci> (\x -> length x > 5) "uhoh"
False
ghci> (\x -> x /= ' ') 'Q'
True
ghci> (\x -> x /= ' ') ' '
False
ghci> filter (\x -> x /= ' ') 'Is the Haskell fun yet?

<interactive>:18:49: error:
    parse error (possibly incorrect indentation or mismatched brackets)
ghci> filter (\x -> x /= ' ') 'Is the Haskell fun yet?'

<interactive>:19:50: error:
    lexical error in string/character literal at end of input
ghci> filter (\x -> x /= ' ') "Is the Haskell fun yet?"
"IstheHaskellfunyet?"
ghci> quit

<interactive>:21:1: error:
    * Variable not in scope: quit
    * Perhaps you meant `quot' (imported from Prelude)
ghci> :quit
Leaving GHCi.

C:\DEV>
```

Task 2

```
squareArea :: Num a => a -> a
squareArea n = n*n

circleArea :: Floating a => a -> a
circleArea n = n*n*pi

blueAreaOfCube :: Floating a => a -> a
blueAreaOfCube n = x-y
  where x = squareArea n * 6
        y = circleArea (n * 0.25) * 6

paintedCube1 n = if n < 3 then 0 else (n-2)*(n-2) * 6

paintedCube2 n = if n < 3 then 0 else (n-2) * 12
```

```
>>> squareArea 10
100
>>> squareArea 12
144
>>> circleArea 10
314.1592653589793
>>> circleArea 12
452.3893421169302
>>> blueAreaOfCube 10
482.19027549038276
>>> blueAreaOfCube 12
694.3539967061512
>>> blueAreaOfCube 1
4.821902754903828
>>> map blueAreaOfCube [1..3]
[4.821902754903828,19.287611019615312,43.39712479413445]
>>> paintedCube1 1
0
>>> paintedCube1 2
0
>>> paintedCube1 3
6
>>> map paintedCube1 [1..10]
[0,0,6,24,54,96,150,216,294,384]
>>> paintedCube2 1
0
>>> paintedCube2 2
0
>>> paintedCube2 3
12
>>> map paintedCube2 [1..10]
[0,0,12,24,36,48,60,72,84,96]
>>>
```

Task 3

```
reverseWords s = (unwords(reverse(words s)))

averageWordLength s =
  fromIntegral(foldl (+) 0 (map length (words s))) / from Integral( length (words s ))
```

```
>>> reverseWords "appa and baby yoda are the best"
"best the are yoda baby and appa"
>>> reverseWords "want me some coffee"
"coffee some me want"
>>> averageWordLength "appa and baby yoda are the best"
3.5714285714285716
>>> averageWordLength "want me some coffee"
4.0
>>>
```

Task 4

```
list2set [] = []
list2set (x:xs) = if (elem x xs) then (list2set xs) else x:list2set xs

isPalindrome [] = True
isPalindrom (x: []) = True
isPalindrome (x:xs) =
  if x == last xs then isPalindrome (init xs)
  else False
```

```
>>> list2set [1,2,3,2,3,4,3,4,5]
[1,2,3,4,5]
>>> list2set "need more coffee"
"ned morcf"
>>> isPalindrome ["coffee","latte","coffee"]
True
>>> isPalindrome ["coffee","latte","espresso","coffee"]
False
>>> isPalindrome [1,2,5,7,11,13,11,7,5,3,2]
False
>>> isPalindrome [2,3,5,7,11,13,11,7,5,3,2]
True
```

Task 5

```
count obj [] = 0
count obj (x:xs) =
if obj == x then 1 + count obj xs
else count obj xs

freqTable list = zip list1 list2
where list1 = [ a | a <- (list2set list) ]
      list2 = [count b list | b <- (list2set list) ]
```

```
>>> count 'e' "need more coffee"
5
>>> count 4 [1,2,3,2,3,4,3,4,5,4,5,6]
3
>>> freqTable "need more coffee"
[('n',1),('e',5),('d',1),(' ',2),('m',1),('o',2),('r',1),('c',1),('f',2)]
>>> freqTable [1,2,3,2,3,4,3,4,5,4,5,6]
[(1,1),(2,2),(3,3),(4,3),(5,2),(6,1)]
>>>
```

Task 6

```
Tgl n = foldl (+) 0 [1..x]

triangleSequence x = mad tri [1..x]

vowelCount st = length (filter (\x -> x == 'a' || x == 'e' || x == 'i' || x == 'o' || x == 'u' ) st)

lcism fun pred list = (map fun (filter pred list))
```

```
>>> tgl 5
15
>>> tgl 10
55
>>> triangleSequence 10
[1,3,6,10,15,21,28,36,45,55]
>>> triangleSequence 20
[1,3,6,10,15,21,28,36,45,55,66,78,91,105,120,136,153,171,190,210]
>>> vowelCount "cat"
1
>>> vowelCount "mouse"
3
>>> lcsim tgl odd [1..15]
[1,6,15,28,45,66,91,120]
```

```
>>> lcsim length (\w -> elem ( head w ) "aeiou") animals
[8,9]
>>>
```

Task 7

```
a :: [Int]
a = [2,5,1,3]
b :: [Int]
b = [1,3,6,2,5]
c :: [Int]
c = [4,4,2,1,1,2,2,4,4,8]
u :: [Int]
u = [2,2,2,2,2,2,2,2,2,2]
x :: [Int]
x = [1,9,2,8,3,7,2,8,1,9]


pairwiseValues :: [Int -> [(Int,Int)]
pairwiseValues (x:[]) = []
pairwiseValues (x:xs) =
[(x,(head xs))]++pairwiseValues xs

pairwiseDifferences :: [Int] -> [Int]
pairwiseDifferences [] = []
pairwiseDifferences list =
map (\(x,y) -> x - y) (pairwiseValues list)

pairwiseSums :: [Int] -> [Int]
pairwiseSums [] = []
pairwiseSums list =
map (\(x,y) -> x + y) (pairwiseValues list)

half :: Int -> Double
half number = ( fromIntegral number ) / 2

pairwiseHalves :: [Int] -> [Double]
pairwiseHalves [] = []
pairwiseHalves list = map half list


pairwiseHalves :: [Int] -> [Double]
pairwiseHalves [] = []
pairwiseHalves list = map half list


pairwiseHalfSums :: [Int] -> [Double]
pairwiseHalfSums list =
pairwiseHalves (pairwiseSums list)
```

```
pairwiseTermPairs :: [Int] -> [(Int, Double )]
pairwiseTermPairs list
zip (pairwiseDifferences list) (pairwiseHalfSums list)

term :: (Int,Double) -> Double
term nPair = abs (fromIntegral (fst nPair)/ (snd nPair))

pairwiseTerms :: [Int] -> [Double]
pairwiseTerms list = map term (pairwiseTermPairs list)


nPVI :: [Int] -> Double
nPVI xs = normalizer xs * sum (pairwiseTerms xs )
 where normalizer xs = 100 / fromIntegral ((length xs ) - 1)
```

```
>>> a
[2, 5, 1, 3]
>>> b
[1, 3, 6, 2, 5]
>>> c
[4, 4, 2, 1, 1, 2, 2, 4, 4, 8]
>>> u
[2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
>>> x
[1, 9, 2, 8, 3 ,7, 2, 8, 1, 9]
```

```
>>> pairwiseValues a
[(2,5),(5,1),(1,3)]
>>> pairwiseValues b
[(1,3),(3,6),(6,2),(2,5)]
>>> pairwiseValues c
[(4,4),(4,2),(2,1),(1,1),(1,2),(2,2),(2,4),(4,4)]
>>> pairwiseValues u
[(2,2),(2,2),(2,2),(2,2),(2,2),(2,2),(2,2),(2,2),(2,2)]
>>> pairwiseValues x
[(1,9),(9,2),(2,8),(8,3),(3,7),(7,2),(2,8),(8,1),(1,9)]
>>>
```

```
>>> pairwiseDifferences a
[-3,4,-2]
>>> pairwiseDifferences b
[-2,-3,4,-3]
>>> pairwiseDifferences c
[0,2,1,0,-1,0,-2,0]
>>> pairwiseDifferences u
[0,0,0,0,0,0,0,0,0]
>>> pairwiseDifferences x
[-8,7,-6,5,-4,5,-6,7,-8]
>>>
```

```
>>> pairwiseSums a
[7,6,4]
>>> pairwiseSums b
[4,9,8,7]
>>> pairwiseSums c
[8,6,3,2,3,4,6,8]
>>> pairwiseSums u
[4,4,4,4,4,4,4,4,4]
>>> pairwiseSums x
[10,11,10,11,10,9,10,9,10]
>>>
```

```
>>> pairwiseHalves [1..10]
[0.5,1.0,1.5,2.0,2.5,3.0,3.5,4.0,4.5,5.0]
>>> pairwiseHalves u
[1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0]
>>> pairwiseHalves x
[0.5,4.5,1.0,4.0,1.5,3.5,1.0,4.0,0.5,4.5]
>>>
```

```
>>> pairwiseHalfSums a
[3.5,3.0,2.0]
>>> pairwiseHalfSums b
[2.0,4.5,4.0,3.5]
>>> pairwiseHalfSums c
[4.0,3.0,1.5,1.0,1.5,2.0,3.0,4.0]
>>> pairwiseHalfSums u
[2.0,2.0,2.0,2.0,2.0,2.0,2.0,2.0,2.0]
>>> pairwiseHalfSums x
[5.0,5.5,5.0,5.5,5.0,4.5,5.0,4.5,5.0]
>>>
```

```
>>> pairwiseTermPairs a
[(-3,3.5),(4,3.0),(-2,2.0)]
>>> pairwiseTermPairs b
[(-2,2.0),(-3,4.5),(4,4.0),(-3,3.5)]
>>> pairwiseTermPairs c
[(0,4.0),(2,3.0),(1,1.5),(0,1.0),(-1,1.5),(0,2.0),(-2,3.0),(0,4.0)]
>>> pairwiseTermPairs u
[(0,2.0),(0,2.0),(0,2.0),(0,2.0),(0,2.0),(0,2.0),(0,2.0),(0,2.0),(0,2.0)]
>>> pairwiseTermPairs x
[(-8,5.0),(7,5.5),(-6,5.0),(5,5.5),(-4,5.0),(5,4.5),(-6,5.0),(7,4.5),(-8,5.0)]
>>>
```

```
>>> pairwiseTerms a
[0.8571428571428571,1.3333333333333333,1.0]
>>> pairwiseTerms b
[1.0,0.6666666666666666,1.0,0.8571428571428571]
>>> pairwiseTerms c
[0.0,0.6666666666666666,0.6666666666666666,0.0,0.6666666666666666,0.0,0.6666666666666666,0.0]
>>> pairwiseTerms u
[0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0]
>>> pairwiseTerms x
[1.6,1.2727272727272727,1.2,0.9090909090909091,0.8,1.1111111111111112,1.2,1.5555555555555556,1.6]
>>>
```

```
>>> nPVI a
106.34920634920636
>>> nPVI b
88.09523809523809
>>> nPVI c
33.333333333333333
>>> nPVI u
0.0
>>> nPVI x
124.98316498316497
>>>
```

Task 8

```
>> dit
"-"
>> dah
"---"
>> dit ++ dah
"----"
>> m
'm',"--- ---")
>> g
'g',"--- --- -")
>> h
'h',"- - - -")
>> symbols
('a',"- ---"),('b',"--- - - -"),('c',"--- - --- -"),('d',"--- - -"),('e',"-"),('f',"- - --- -"),('g',"--- --- -"),('h',
'- - - -"),('i',"- -"),('j',"- --- --- ---"),('k',"--- - ---"),('l',"- --- - -"),('m',"--- ---"),('n',"--- -"),('o',"---
--- ---"),('p',"- --- --- -"),('q',"--- --- - ---"),('r',"- --- -"),('s',"- - -"),('t',"---"),('u',"- - ---"),('v',"- -
- ---"),('w',"- --- ---"),('x',"--- - - ---"),('y',"--- - --- ---"),('z',"--- --- - -")]
>>
```

```
>>> assoc 'x' symbols
('x', "--- - - ---")
>>> assoc 'y' symbols
('y', "--- - --- ---")
>>> find 'w'
"- --- ---"
>>> find 'z'
"--- --- - -"
>>>
```

```
>>> addletter "x" "-- - -"
"x   -- - -"
>>> addword "good" "----- -- --"
"good      ----- -- --"
>>> droplast3 "good"
"g"
>>> droplast7 "this is a test"
"this is"
>>>
```

```
>>> encodeletter 'm'
"--- ---"
>>> encodeletter 'x'
"--- - - ---"
>>> encodeletter 'y'
"--- - --- ---"
>>> encodeword "yay"
"--- - --- ---   - ---   --- - --- ---"
>>> encodeword "nay"
"--- -   - ---   --- - --- ---"
>>> encodeword "test"
"---   -   - - -   ---"
>>> encodemessage "need more coffee"
"--- -   -   -   --- - -     --- ---   --- --- ---   - --- -   -     --- - --- -   --- --- ---   - - --- -   - - ---
 -   -   -"
>>> encodemessage "secret message here"
"- - -   -   --- - --- -   - --- -   -   ---     --- ---   -   - - -   - - -   - ---   --- --- -   -     - - - -   -
   - --- -   -"
>>> encodemessage "impossible to crack"
"- -   --- ---   - --- --- -   --- --- ---   - - -   - - -   - -   --- - - -   - --- - -   -     ---   --- --- ---
  --- - --- -   - --- -   - ---   --- - --- -   --- - ---"
>>>
```