Joseph M. Scollo

Haskell Programming Assignment: Various Computations.

Abstract:

Various computations in haskell that demonstrate recursion, higher order functions, list processing. This assignment has a an array of tasks that gets one acquainted with functional programming and haskell syntax.

Task 1: Mindfully Mimic the Demo

Demo

```
Prelude> :set prompt ">>>"
>>>length [2,3,5,7]
4
>>>words "need more coffee"
["need","more","coffee"]
>>>unwords ["need","more","coffee"]
"need more coffee"
>>>reverse "need more coffee"
"eeffoc erom deen"
>>>reverse ["need","more","coffee"]
["coffee","more","need"]
>>>head ["need","more","coffee"]
"need"
>>>tail ["need","more","coffee"]
["more","coffee"]
>>>last ["need","more","coffee"]
"coffee"
>>>init ["need","more","coffee"]
["need","more"]
>>>take 7 ["need","more","coffee"]
["need","more","coffee"]
>>>take 7 "need more coffee"
"need mo"
>>>drop 7 "need more coffee"
"re coffee"
>>>( \x -> length x > 5) "Friday"
True
>>>( \x -> length x > 5) "uhoh"
False
>>>( \x -> x /= ' ') 'Q'
True
>>>( \x -> x /= ' ') ' '
False
>>>filter( \x -> x /= ' ') "Is the Haskell fun yet?"
"IstheHaskellfunyet?"
>>>:quit
```

Task 2 – Numeric Function Definitions:

Code

```
squareArea side = side * side

circleArea radius = pi * (radius * radius)

blueAreaOfCube side = (6*((squareArea side) - (circleArea (side / 4))))

paintedCube1 cube = 6 * ((cube - 2) * (cube - 2))

paintedCube2 cube = 12 * (cube - 2)
```

Demo:

```
Ok, one module loaded.
*Main> :set prompt ">>>"
>>>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
6
>>>paintedCube1 2
0
>>>paintedCube1 3
6
>>>map paintedCube1 [1..10]
[6,0,6,24,54,96,150,216,294,384]
>>>paintedCube2 1
-12
>>>paintedCube2 2
0
>>>paintedCube2 3
12
>>>map paintedCube2 [1..10]
[-12,0,12,24,36,48,60,72,84,96]
>>>:quit
Leaving GHCi.
```

Task 3 – Puzzlers

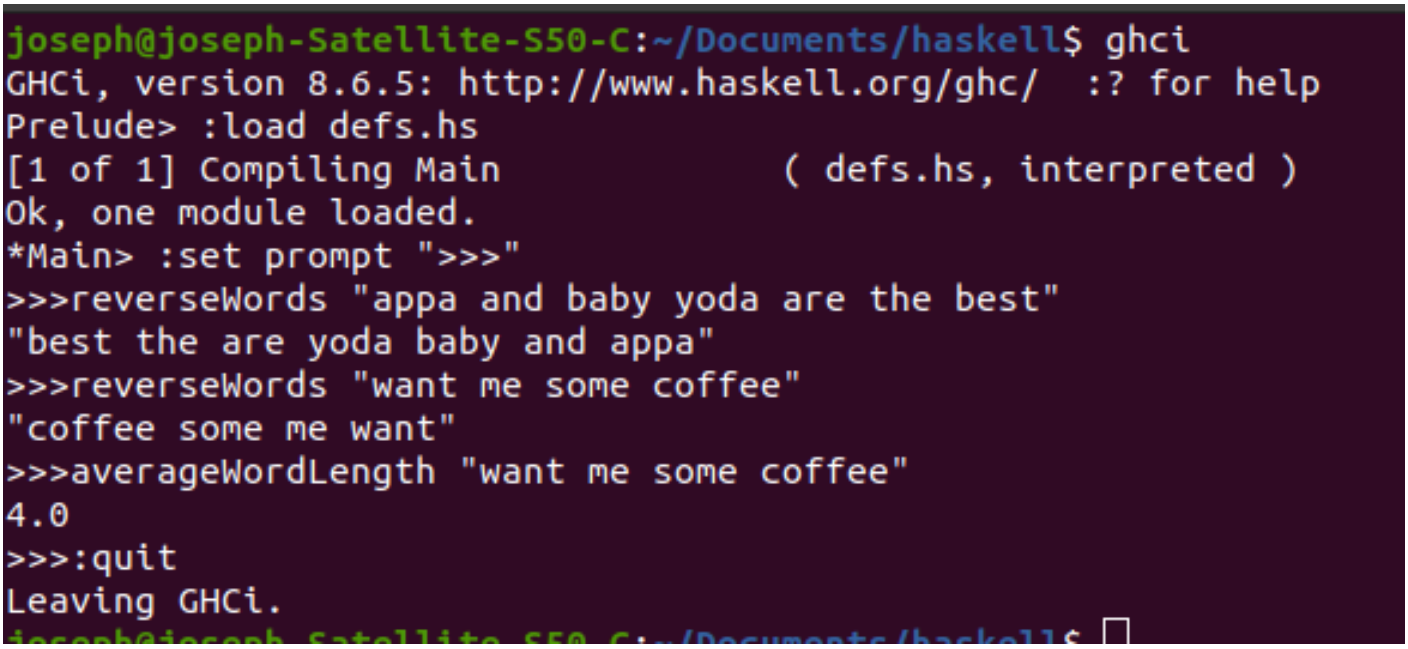
code

```haskell
reverseWords :: String -> String
reverseWords w = unwords (reverse (words w))

averageWordLength :: String -> Float
averageWordLength w = (fromIntegral (sum (map length (words w)))) / (fromIntegral (length (words w)))

```

Demo



Task 4 – Recursive List Processors:

Code:

```haskell
list2set lst = if (length lst) == 0
 then [] else (if elem (head lst) set then set else (head lst) : set)
 where set = list2set(tail lst)

isPalindrome w = if (length w) < 2 then True
  else (if (head w) == (last w)
  then (isPalindrome (tail (init w)))
  else False)

colatz :: Int -> [Int]
colatz n = if (even n) then n: colatz (n `div` 2)
 else if (n == 1) then [1] else n:colatz (3 * n + 1)
```

Demo:

```
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :load defs.hs
[1 of 1] Compiling Main             ( defs.hs, interpreted )
Ok, one module loaded.
*Main> :set prompt: ">>>"
Some flags have not been recognized: prompt:, >>>
*Main> :set prompt ">>>"
>>>list2set [1,2,3,2,3,4,3,4,5]
[1,2,3,4,5]
>>>list2set "need more coffee"
"ndmr cofe"
>>>isPalindrome ["coffee","latte","coffee"]
True
>>>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
>>>colatz 10
[10,5,16,8,4,2,1]
>>>colatz 11
[11,34,17,52,26,13,40,20,10,5,16,8,4,2,1]
>>>colatz 100
[100,50,25,76,38,19,58,29,88,44,22,11,34,17,52,26,13,40,20,10,5,16,8,4,2,1]
>>>
>>>:quit
Leaving GHCi.
```

Task 5 – List Comprehensions:

Code

```
32 count n l = length [x | x <- l , n == x]
33 freqTable l = zip(list2set l) (map(\x -> count x l) (list2set l))
34
```

Demo

```
joseph@joseph-Satellite-S50-C:~/Documents/haskell$ ghci
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :load defs.hs
[1 of 1] Compiling Main             ( defs.hs, interpreted )
Ok, one module loaded.
*Main> :set prompt ">>>"
>>>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),('d',1),('m',1),('r',1),(' ',2),('c',1),('o',2),('f',2),('e',5)]
>>>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)]
>>>:quit
Leaving GHCi.
joseph@joseph-Satellite-S50-C:~/Documents/haskell$
```

Task 6 – Higher Order Functions

Code

```
35 tgl x = foldl (+) 0 [1..x]
36
37 triangleSequence ts = map tgl [1..ts]
38
39 vowelCount s = length v
40  where v = filter (\x -> elem x "a e i o u") s
41
42 lcsim fn p l = map fn fl
43  where fl = filter p l
44
```

-

Demo

```
joseph@joseph-Satellite-S50-C:~/Documents/haskell$ ghci
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :load defs.hs
[1 of 1] Compiling Main             ( defs.hs, interpreted )
Ok, one module loaded.
*Main> :set prompt ">>>"
>>>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]
>>>animals = ["elephant","lion","tiger","orangatan","jaguar"]
>>>lcsim length (\w -> elem (head w) "aeiou") animals
[8,9]
>>>
```

Task 7- An Interesting Statistic: nPVI:

Test Data

```
45  -- Test data --
46
47 a :: [Int]
48 a = [2,5,1,3]
49
50 b :: [Int]
51 b = [1,3,6,2,5]
52
53 c :: [Int]
54 c = [1,9,8,2]
55
56 u :: [Int]
57 u = [2,2,2,2,2,2,2,2,2,2]
58
59 x :: [Int]
60 x = [1,9,2,8,3,7,2,8,1,9]
61
```

Task 7b: The pairwiseValues Function:

code

```
61
62 pairwiseValues :: [Int] -> [(Int,Int)]
63 pairwiseValues xs = zip (init xs) (tail xs)
64
```

DEMO:

```
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :load defs.hs
[1 of 1] Compiling Main             ( defs.hs, interpreted )
Ok, one module loaded.
*Main> :set prompt ">>>"
>>>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),(4,8)]
>>>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)]
>>>:quit
Leaving GHCi.
joseph@joseph-Satellite-S50-C:~/Documents/haskell$
```

Task 7c – The PairwiseDifferences

Code

```
65 pairwiseDifferences :: [Int] -> [Int]
66 pairwiseDifferences e = map (\(x,y) -> x - y ) (pairwiseValues e)
67
```

Demo

```
joseph@joseph-Satellite-S50-C:~/Documents/haskell$ ghci
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :load defs.hs
[1 of 1] Compiling Main             ( defs.hs, interpreted )
Ok, one module loaded.
*Main> :set prompt ">>>"
>>>pairwiseDifferences a
[-3,4,-2]
>>>pairwiseDifferences b
[-2,-3,4,-3]
>>>pairwiseDifferences c
[0,2,1,0,-1,0,-2,0,-4]
>>>pairwiseDifferences u
[0,0,0,0,0,0,0,0,0]
>>>pairwiseDifferences x
[-8,7,-6,5,-4,5,-6,7,-8]
>>>:quit
Leaving GHCi.
```

Task 7d – Pairwise Sums:

Code:

```
68 pairwiseSums :: [Int] -> [Int]
69 pairwiseSums e = map (\(x,y) -> x + y ) (pairwiseValues e)
70
```

–

Demo

```
joseph@joseph-Satellite-S50-C:~/Documents/haskell$ ghci
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :load defs.hs
[1 of 1] Compiling Main             ( defs.hs, interpreted )
Ok, one module loaded.
*Main> :set prompt ">>>"
>>>pairwiseSums a
[7,6,4]
>>>pairwiseSums b
[4,9,8,7]
>>>pairwiseSums c
[8,6,3,2,3,4,6,8,12]
>>>pairwiseSums u
[4,4,4,4,4,4,4,4,4]
>>>pairwiseSums x
[10,11,10,11,10,9,10,9,10]
>>>:quit
Leaving GHCi.
joseph@joseph-Satellite-S50-C:~/Documents/haskell$
```

Task 7e – PairwiseHalves Functions

Demo:

```
72 half :: Int -> Double
73 half n = (fromIntegral n) / 2
74
75 pairwiseHalves :: [Int] -> [Double]
76 pairwiseHalves n = map (half) n
```

Demo

```
joseph@joseph-Satellite-S50-C:~/Documents/haskell$ ghci
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :load defs.hs
[1 of 1] Compiling Main             ( defs.hs, interpreted )
Ok, one module loaded.
*Main> :set prompt ">>>"
>>>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]
>>>:quit
Leaving GHCi.
joseph@joseph-Satellite-S50-C:~/Documents/haskell$
```

Task 7f – pairwiseHalfSums:

Code:

```
..
78 pairwiseHalfSums :: [Int] -> [Double]
79 pairwiseHalfSums n = pairwiseHalves (pairwiseSums n)
80
```

-

Demo

```
joseph@joseph-Satellite-S50-C:~/Documents/haskell$ ghci
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :load defs.hs
[1 of 1] Compiling Main             ( defs.hs, interpreted )
Ok, one module loaded.
*Main> :set prompt ">>>"
>>>pairwiseHalfSums a
[3.5,3.0,2.0]
>>>pairwiseHalfSums b
[2.0,4.5,4.0,3.5]
>>>pairwiseHalfSums c
[4.0,3.0,1.5,1.0,1.5,2.0,3.0,4.0,6.0]
>>>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]
>>>:quit
Leaving GHCi.
joseph@joseph-Satellite-S50-C:~/Documents/haskell$
```

Task 7g - pairwisTermPairs

Code:

```
80
81 pairwiseTermPairs :: [Int] -> [(Int,Double)]
82 pairwiseTermPairs n = zip (pairwiseDifferences n) (pairwiseHalfSums n)
83
```

Demo

```
joseph@joseph-Satellite-S50-C:~/Documents/haskell$ ghci
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :load defs.hs
[1 of 1] Compiling Main             ( defs.hs, interpreted )
Ok, one module loaded.
*Main> :set prompt ">>>"
>>>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),(-4,6.0)]
>>>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)]
>>>:quit
Leaving GHCi.
```

Task 7h – pairwiseTerms:

Code

```haskell
term :: (Int,Double) -> Double
term prs = abs (fromIntegral ( fst prs ) / ( snd prs ) )

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

-

Demo

```
joseph@joseph-Satellite-S50-C:~/Documents/haskell$ ghci
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :load defs.hs
[1 of 1] Compiling Main             ( defs.hs, interpreted )
Ok, one module loaded.
*Main> :set prompt ">>>"
>>>pairwiseTerms a
[0.8571428571428571,1.3333333333333333,1.0]
>>>pairwiseTerms b
[1.0,0.6666666666666666,1.0,0.8571428571428571]
>>>pairwiseTerms c
[0.0,0.6666666666666666,0.6666666666666666,0.0,0.6666666666666666,0.0,0.6666666666666666,0.0,0.6666666666666666]
>>>pairwiseTerms u
[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]
>>>:quit
Leaving GHCi.
```

Task 7i – nPVI Function:

Code

```
89
90 nPVI :: [Int] -> Double
91 nPVI xs = normalizer xs * sum ( pairwiseTerms xs )
92  where normalizer xs = 100 / fromIntegral ( (length xs ) - 1 )
93
```

-

Demo

```
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :load defs.hs
[1 of 1] Compiling Main             ( defs.hs, interpreted )
Ok, one module loaded.
*Main> :set prompt ">>>"
>>>nPVI a
106.34920634920636
>>>nPVI b
88.09523809523809
>>>nPVI c
37.03703703703703
>>>nPVI u
0.0
>>>nPVI x
124.98316498316497
>>>:quit
Leaving GHCi.
joseph@joseph-Satellite-S50-C:~/Documents/haskell$
```

Task 8 – Historic code: The Dit Dah code:

Subtask 8a:

-

```
Prelude> :load ditdah.hs
[1 of 1] Compiling Main             ( ditdah.hs, interpreted )
Ok, one module loaded.
*Main> :set prompt ">>>"
>>>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 a "- ---"
```

Subtask 8b:

```
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :load ditdah.hs
[1 of 1] Compiling Main             ( ditdah.hs, interpreted )
Ok, one module loaded.
*Main> :set prompt ">>>"
>>>assoc 'a' symbols
('a',"- ---")
>>>assoc 'z' symbols
('z',"--- --- - -")
>>>find 'z'
"--- --- - -"
>>>find 'l'
"- --- - -"
>>>\n
```

Subtask 8c:

```
joseph@joseph-Satellite-S50-C:~/Documents/haskell$ ghci
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :load ditdah.hs
[1 of 1] Compiling Main             ( ditdah.hs, interpreted )
Ok, one module loaded.
*Main> :set prompt ">>>"
>>>addletter (encodeletter 'j') (encodeletter 'o')
"- --- --- ---   --- --- ---"
>>>addword (encodeword "star") (encodeword "wars")
"- - -   ---   - ---   - --- -       - --- ---   - ---   - --- -   - - -"
>>>droplast3 [1,2,3,4,5]
[1,2]
>>>droplast5 [5,4,3,2,1]
```

```
>>>droplast3 [5,4,3,2,1]
[5,4]
>>>droplast7 [5,4,3,2,1]
[]
>>>:quit
Leaving GHCi.
```

Subtask 8d:

```
joseph@joseph-Satellite-S50-C:~/Documents/haskell$ ghci
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :load ditdah.hs
[1 of 1] Compiling Main             ( ditdah.hs, interpreted )
Ok, one module loaded.
*Main> :set prompt ">>>"
>>>encodeletter 'm'
"--- ---"
>>>encodeletter 'a'
".- ..."
>>>encodeletter 'p'
".- --- --- ."
>>>encodeword "yay"
"--- .- --- ---   -.--   --- .- --- ---"
>>>encodeword "ice"
".. .   --- .- --- .-   ."
>>>encodeword "cream"
"--- .- --- .-   .- --- .-   .-   .- ...   --- ---"
>>>encodemessage "need more coffee"
"--- .-   -.   .-   --- .- .-      --- ---   --- --- ---   .- --- .-   .-      --- .- --- .-   --- --- ---   .- .- --- .-   .- .- --- .-   .-   ."
>>>encodemessage "need way more coffee"
"--- .-   -.   .-   --- .- .-      .- --- ---   .- ...   --- .- --- ---      --- ---   --- --- ---   .- --- .-   .-      --- .- --- .-   --- --- ---   .- .- --- .-   .- .- --- .-   .-   ."
>>>encodemessage "there is not enough coffee in the world"
"...   - . - .   -.   . --- .-   .-      .- .-   .- .-      --- .-   --- --- ---   ---      .-   --- .-   --- --- ---   .- .- ---   --- --- .-   .- .- .-   --- .- --- .-   --- --- ---   .- .- --- .-   .- .- --- .-   .- .-   .-      .- .-   --- .-      ---   .- .- .-   .-      .- --- ---   --- --- ---   .- --- .-   .- --- .-   .-   --- .- ."
>>>:quit
```