Temitope Emokpae 5/5/2023

CSC 344

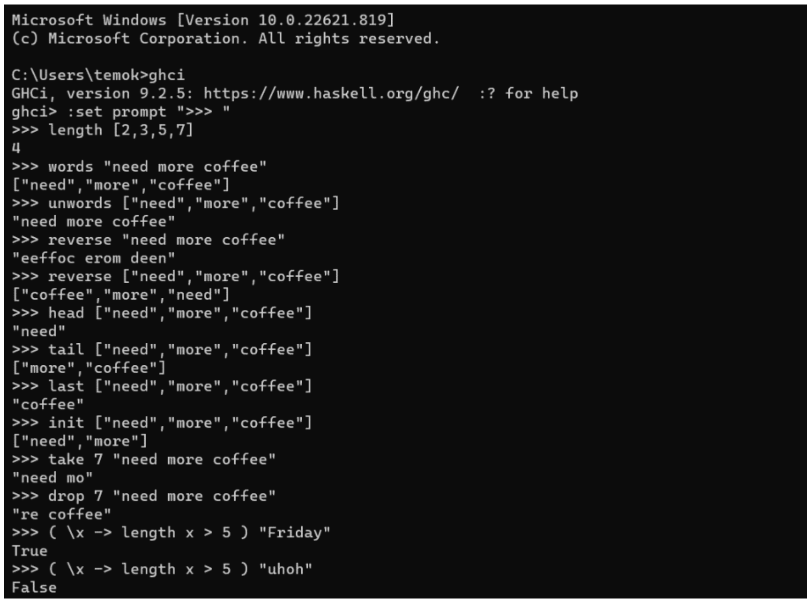
Haskell Programming Assignment: Various Computations

Learning Abstract

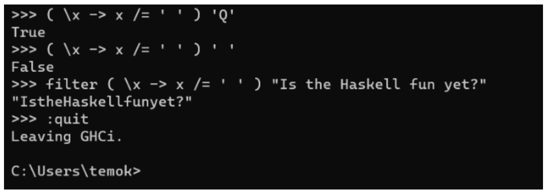
This programming assignment is about computational functions in Haskell. It involves tasks that focus on mimicking functions, recursive list processing, list comprehensions, and higher order functions.

Task 1 - Mindfully Mimicking the Demo

Demo:



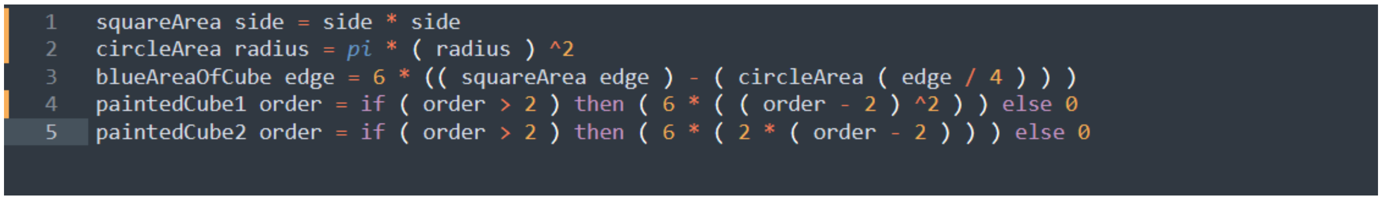
```
Microsoft Windows [Version 10.0.22621.819]
(c) Microsoft Corporation. All rights reserved.

C:\Users\temok>ghci
GHCi, version 9.2.5: https://www.haskell.org/ghc/  :? for help
ghci> :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 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
Leaving GHCi.

C:\Users\temok>
```

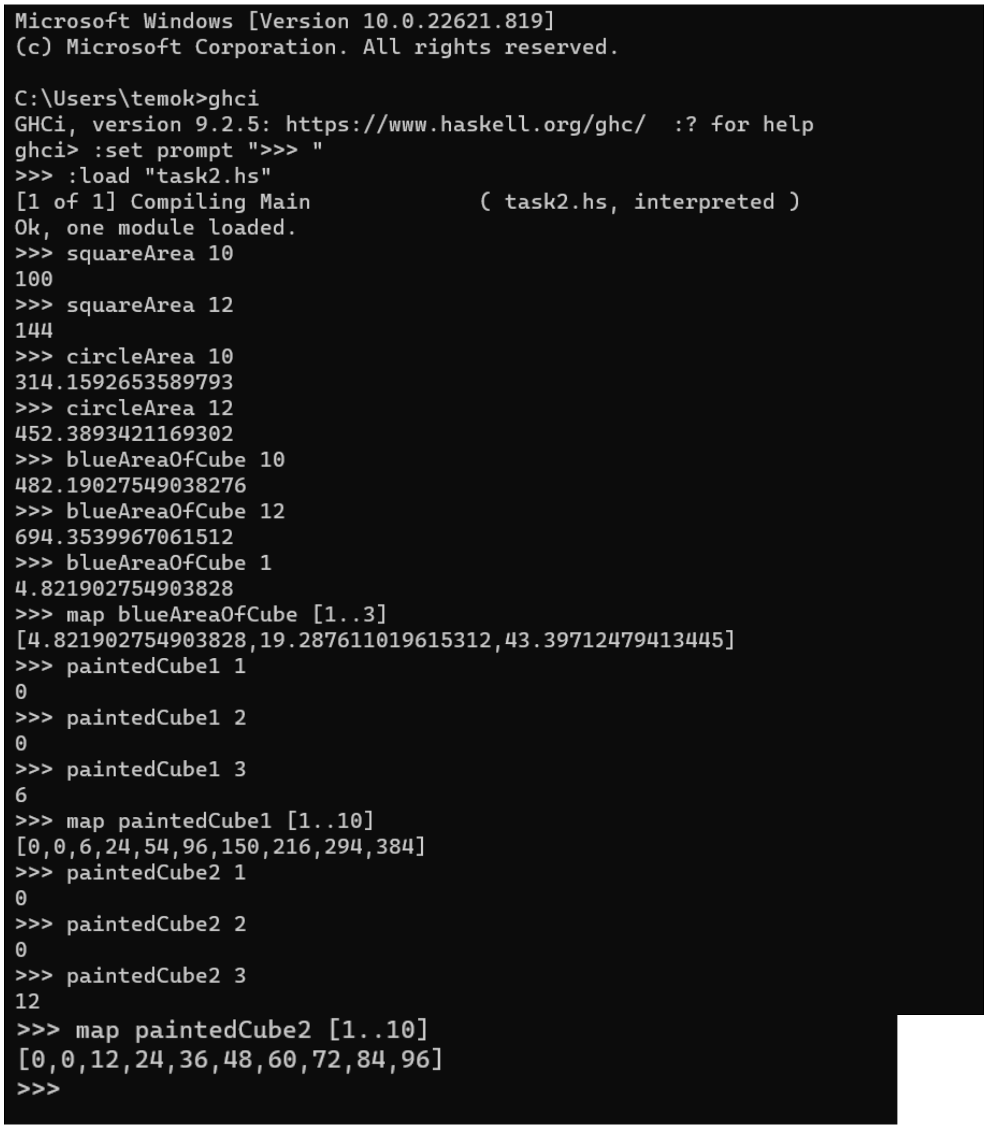
Task 2 - Numeric Function Definitions

Code:



```haskell
squareArea side = side * side
circleArea radius = pi * ( radius ) ^2
blueAreaOfCube edge = 6 * (( squareArea edge ) - ( circleArea ( edge / 4 ) ) )
paintedCube1 order = if ( order > 2 ) then ( 6 * ( ( order - 2 ) ^2 ) ) else 0
paintedCube2 order = if ( order > 2 ) then ( 6 * ( 2 * ( order - 2 ) ) ) else 0
```

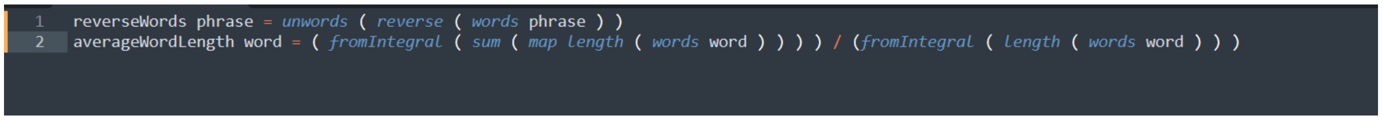
Demo:



```
Microsoft Windows [Version 10.0.22621.819]
(c) Microsoft Corporation. All rights reserved.

C:\Users\temok>ghci
GHCi, version 9.2.5: https://www.haskell.org/ghc/  :? for help
ghci> :set prompt ">>> "
>>> :load "task2.hs"
[1 of 1] Compiling Main             ( task2.hs, interpreted )
Ok, one module loaded.
>>> 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]
>>>
```

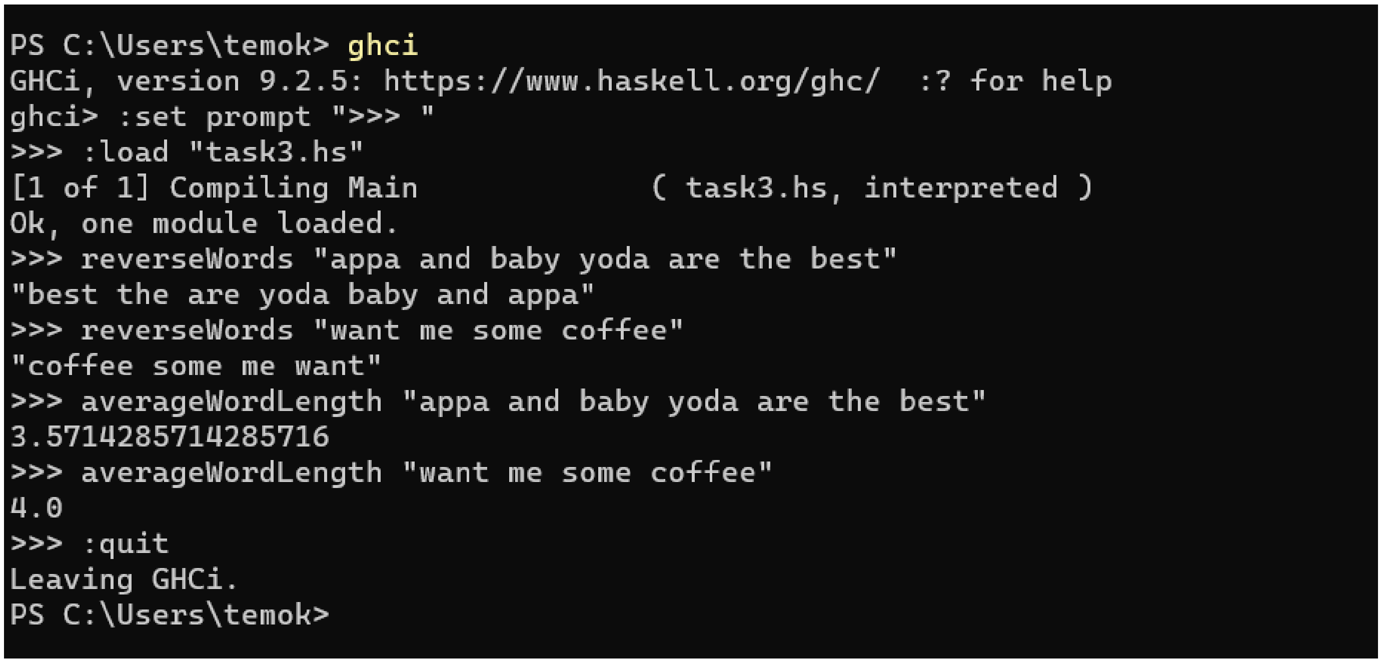
Task 3 - Puzzlers

Code:



```haskell
reverseWords phrase = unwords ( reverse ( words phrase ) )
averageWordLength word = ( fromIntegral ( sum ( map length ( words word ) ) ) ) / (fromIntegral ( length ( words word ) ) )
```

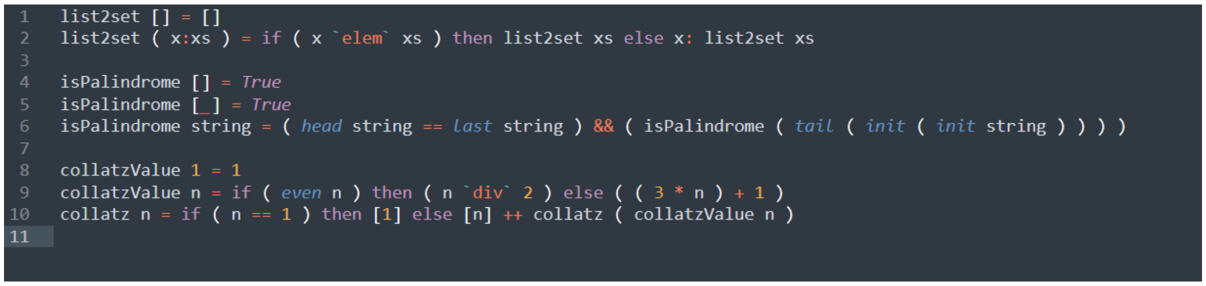
Demo:



```
PS C:\Users\temok> ghci
GHCi, version 9.2.5: https://www.haskell.org/ghc/  :? for help
ghci> :set prompt ">>> "
>>> :load "task3.hs"
[1 of 1] Compiling Main             ( task3.hs, interpreted )
Ok, one module loaded.
>>> 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
>>> :quit
Leaving GHCi.
PS C:\Users\temok>
```

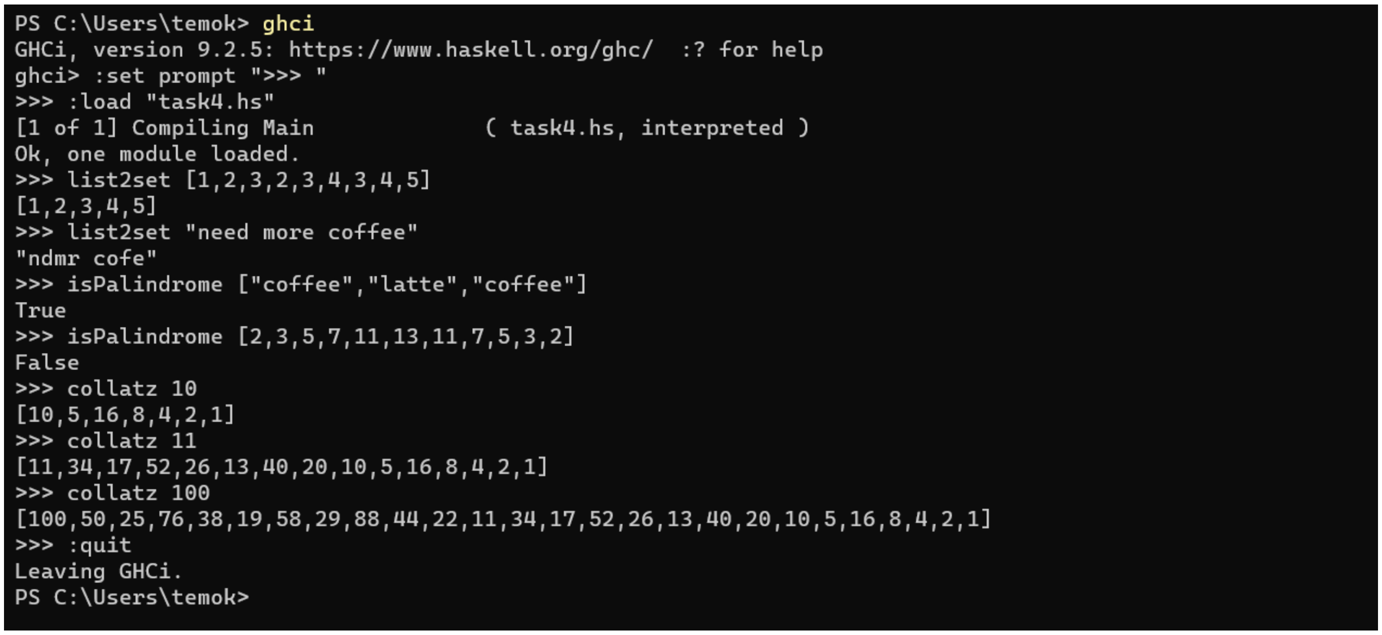
Task 4 - Recursive List Processors

Code:



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

isPalindrome [] = True
isPalindrome [_] = True
isPalindrome string = ( head string == last string ) && ( isPalindrome ( tail ( init ( init string ) ) ) )

collatzValue 1 = 1
collatzValue n = if ( even n ) then ( n `div` 2 ) else ( ( 3 * n ) + 1 )
collatz n = if ( n == 1 ) then [1] else [n] ++ collatz ( collatzValue n )

```

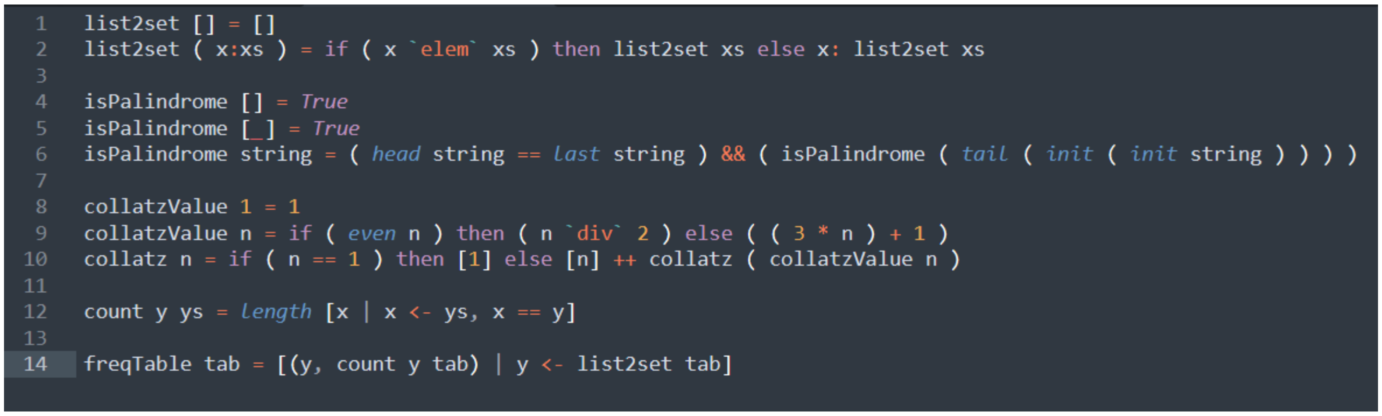
Demo:



```
PS C:\Users\temok> ghci
GHCi, version 9.2.5: https://www.haskell.org/ghc/  :? for help
ghci> :set prompt ">>> "
>>> :load "task4.hs"
[1 of 1] Compiling Main             ( task4.hs, interpreted )
Ok, one module loaded.
>>> 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 [2,3,5,7,11,13,11,7,5,3,2]
False
>>> collatz 10
[10,5,16,8,4,2,1]
>>> collatz 11
[11,34,17,52,26,13,40,20,10,5,16,8,4,2,1]
>>> collatz 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.
PS C:\Users\temok>
```

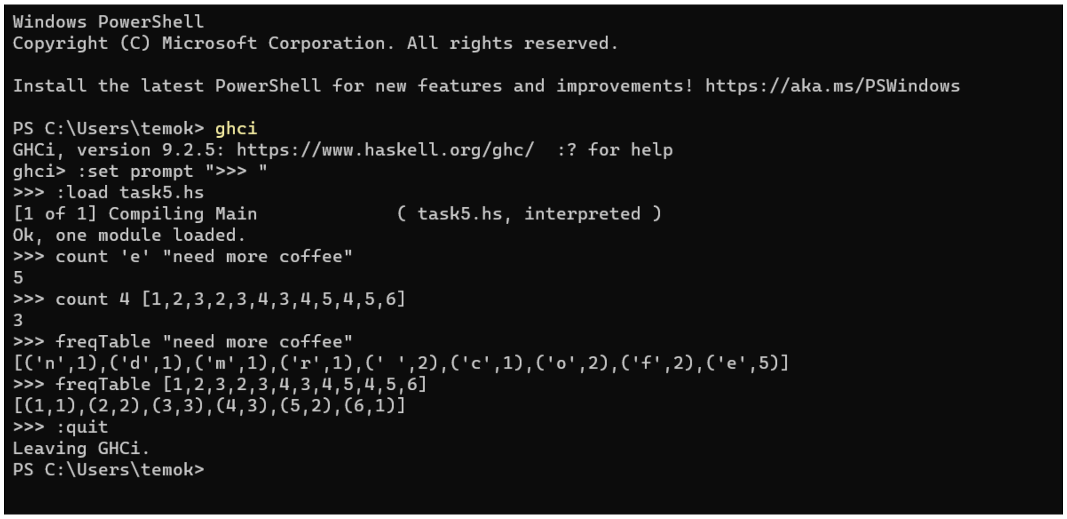
Task 5 - List Comprehensions

Code:



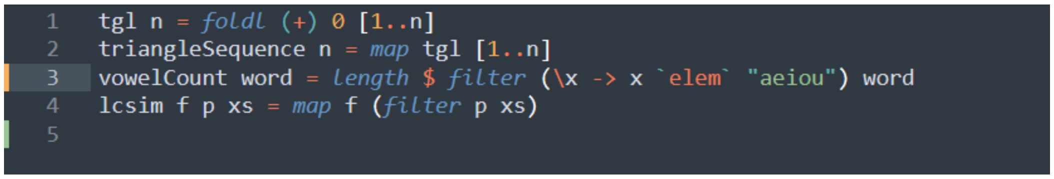
```haskell
list2set [] = []
list2set ( x:xs ) = if ( x `elem` xs ) then list2set xs else x: list2set xs

isPalindrome [] = True
isPalindrome [_] = True
isPalindrome string = ( head string == last string ) && ( isPalindrome ( tail ( init ( init string ) ) ) )

collatzValue 1 = 1
collatzValue n = if ( even n ) then ( n `div` 2 ) else ( ( 3 * n ) + 1 )
collatz n = if ( n == 1 ) then [1] else [n] ++ collatz ( collatzValue n )

count y ys = length [x | x <- ys, x == y]

freqTable tab = [(y, count y tab) | y <- list2set tab]
```

Demo:

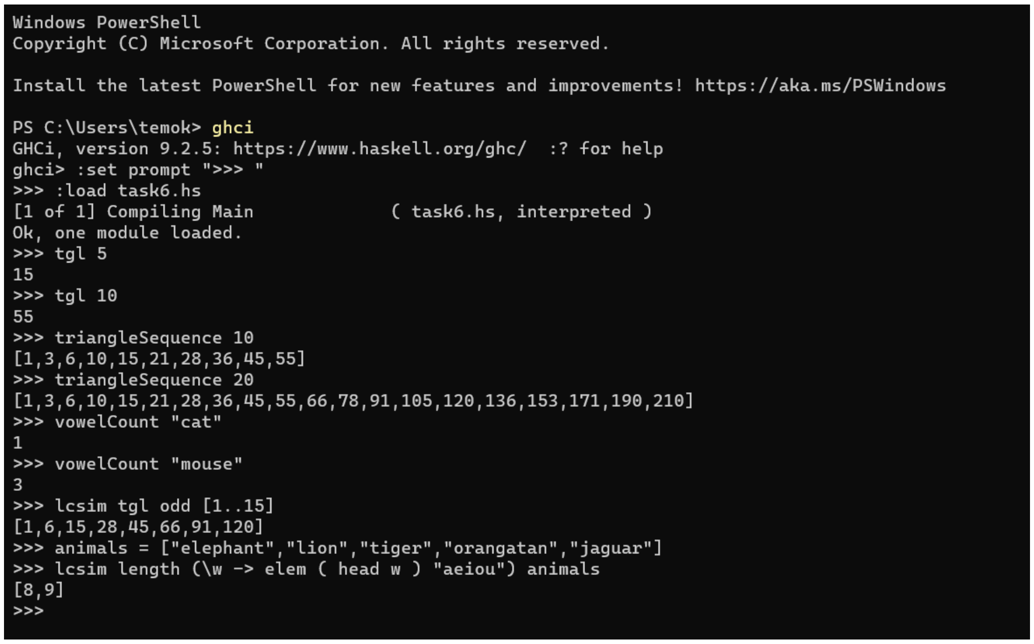

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\temok> ghci
GHCi, version 9.2.5: https://www.haskell.org/ghc/  :? for help
ghci> :set prompt ">>> "
>>> :load task5.hs
[1 of 1] Compiling Main             ( task5.hs, interpreted )
Ok, one module loaded.
>>> 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.
PS C:\Users\temok>
```

Task 6 - Higher Order Functions

Code:

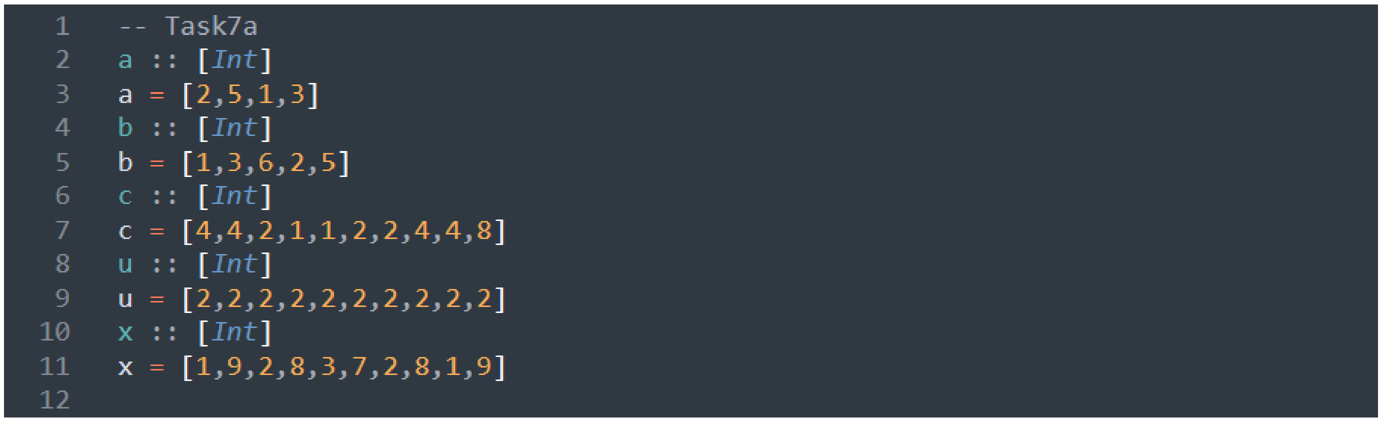

```haskell
tgl n = foldl (+) 0 [1..n]
triangleSequence n = map tgl [1..n]
vowelCount word = length $ filter (\x -> x `elem` "aeiou") word
lcsim f p xs = map f (filter p xs)
```

Demo:


```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\temok> ghci
GHCi, version 9.2.5: https://www.haskell.org/ghc/  :? for help
ghci> :set prompt ">>> "
>>> :load task6.hs
[1 of 1] Compiling Main             ( task6.hs, interpreted )
Ok, one module loaded.
>>> 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]
>>>
```

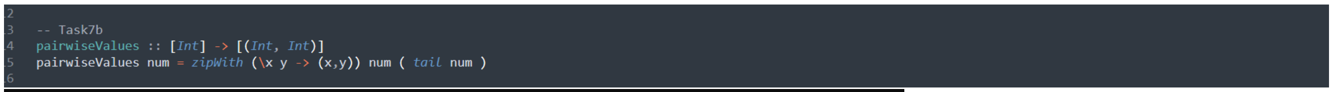
Task 7 - An Interesting Statistic: nPVI

Task 7a:

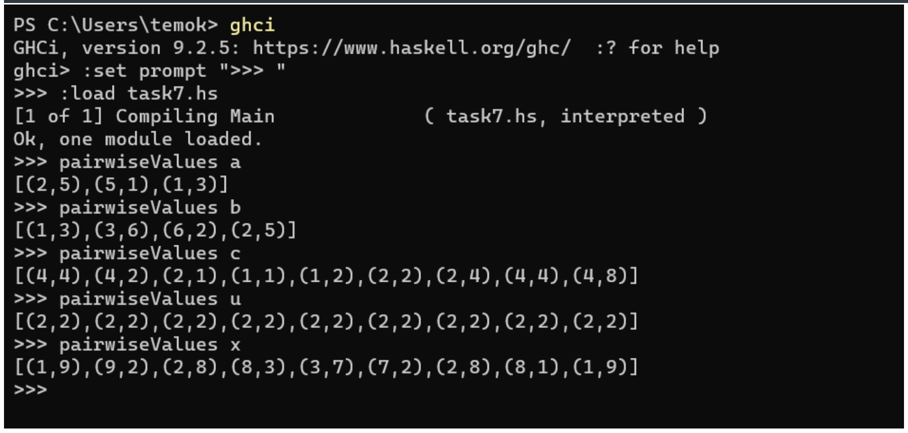


```haskell
-- Task7a
a :: [Int]
a = [2,5,1,3]
b :: [Int]
b = [1,3,6,2,5]
c :: [Int]
c = [4,4,2,1,1,2,2,4,4,8]
u :: [Int]
u = [2,2,2,2,2,2,2,2,2,2]
x :: [Int]
x = [1,9,2,8,3,7,2,8,1,9]
```

Task 7b:

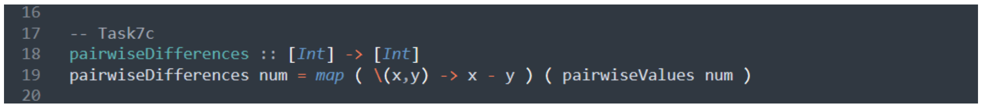


```haskell
-- Task7b
pairwiseValues :: [Int] -> [(Int, Int)]
pairwiseValues num = zipWith (\x y -> (x,y)) num ( tail num )
```

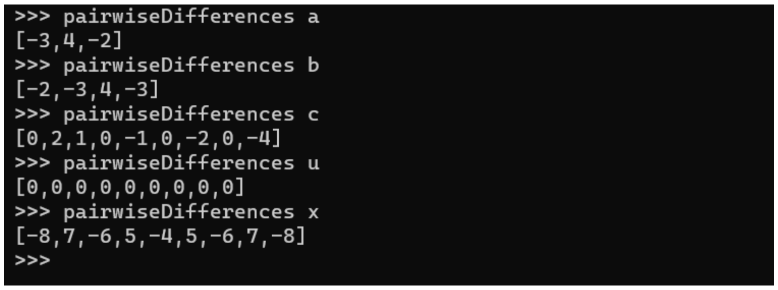


```
PS C:\Users\temok> ghci
GHCi, version 9.2.5: https://www.haskell.org/ghc/  :? for help
ghci> :set prompt ">>> "
>>> :load task7.hs
[1 of 1] Compiling Main             ( task7.hs, interpreted )
Ok, one module loaded.
>>> pairwiseValues a
[(2,5),(5,1),(1,3)]
>>> pairwiseValues b
[(1,3),(3,6),(6,2),(2,5)]
>>> 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)]
>>>
```

Task 7c:

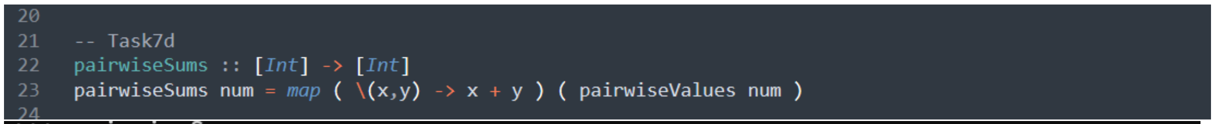


```haskell
-- Task7c
pairwiseDifferences :: [Int] -> [Int]
pairwiseDifferences num = map ( \(x,y) -> x - y ) ( pairwiseValues num )
```

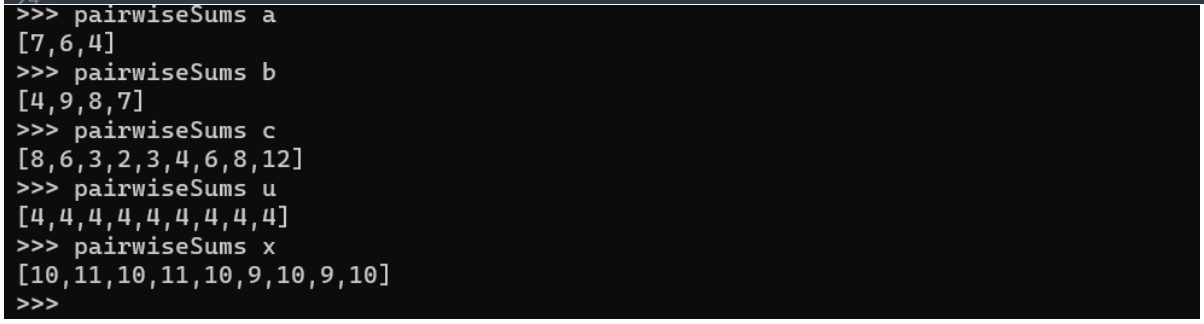


```
>>> pairwiseDifferences a
[-3,4,-2]
>>> pairwiseDifferences b
[-2,-3,4,-3]
>>> pairwiseDifferences c
[0,2,1,0,-1,0,-2,0,-4]
>>> pairwiseDifferences u
[0,0,0,0,0,0,0,0,0]
>>> pairwiseDifferences x
[-8,7,-6,5,-4,5,-6,7,-8]
>>>
```

Task 7d:

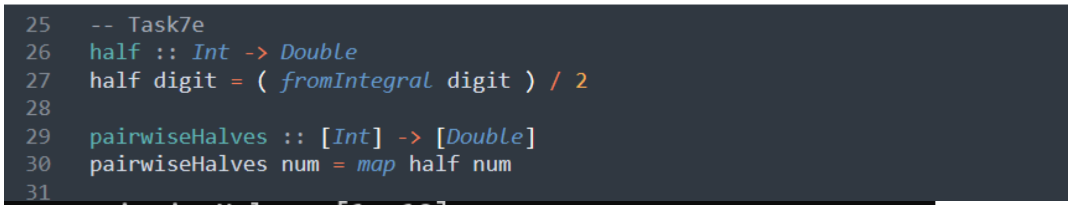


```haskell

-- Task7d
pairwiseSums :: [Int] -> [Int]
pairwiseSums num = map ( \(x,y) -> x + y ) ( pairwiseValues num )
```

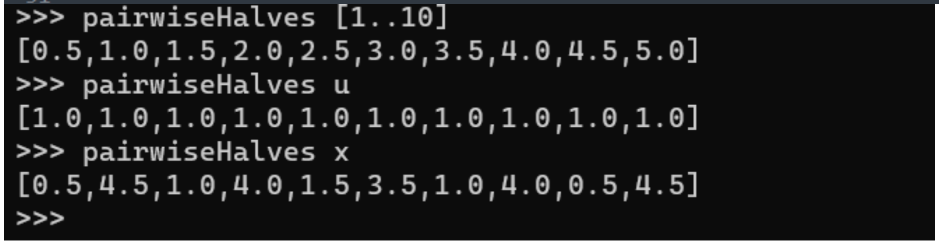


```
>>> pairwiseSums a
[7,6,4]
>>> pairwiseSums b
[4,9,8,7]
>>> pairwiseSums c
[8,6,3,2,3,4,6,8,12]
>>> pairwiseSums u
[4,4,4,4,4,4,4,4,4]
>>> pairwiseSums x
[10,11,10,11,10,9,10,9,10]
>>>
```

Task 7e:

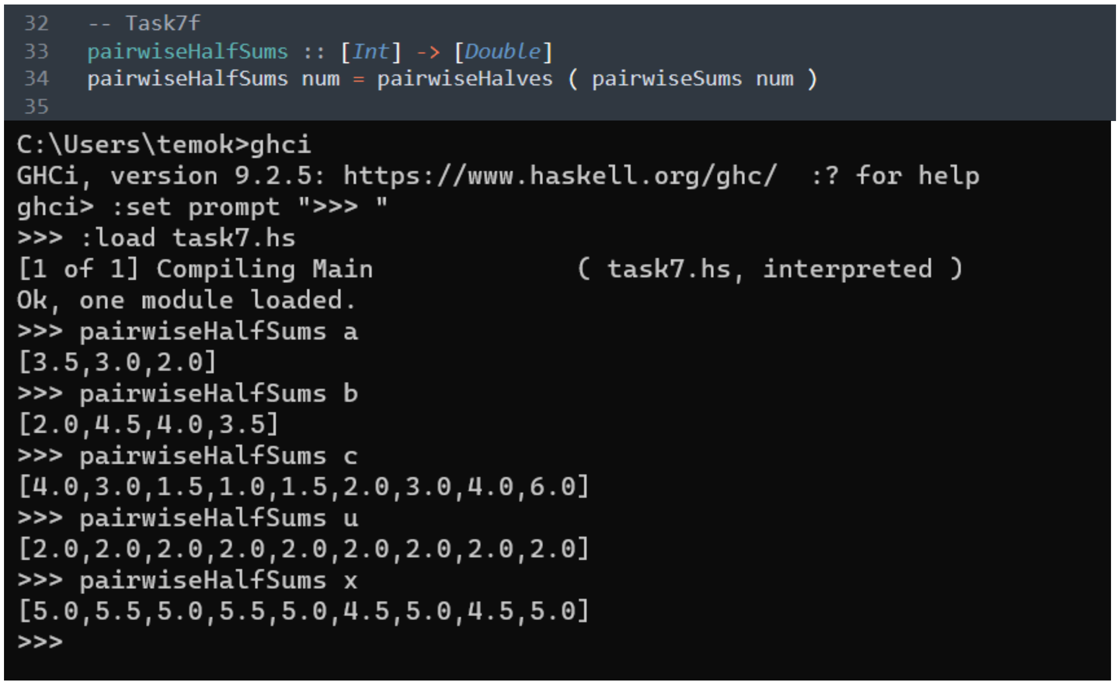


```haskell
-- Task7e
half :: Int -> Double
half digit = ( fromIntegral digit ) / 2

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

```



```
>>> 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]
>>>
```

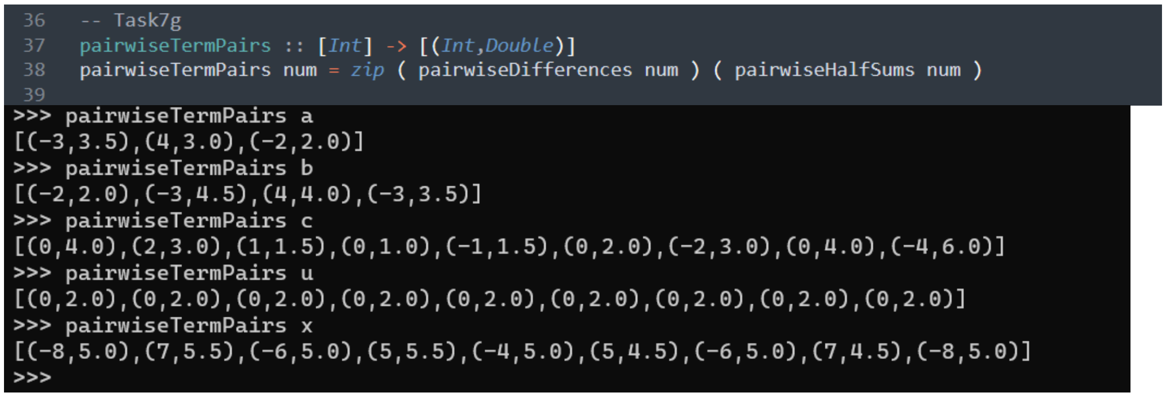
Task 7f:



```haskell
-- Task7f
pairwiseHalfSums :: [Int] -> [Double]
pairwiseHalfSums num = pairwiseHalves ( pairwiseSums num )
```

```
C:\Users\temok>ghci
GHCi, version 9.2.5: https://www.haskell.org/ghc/  :? for help
ghci> :set prompt ">>> "
>>> :load task7.hs
[1 of 1] Compiling Main             ( task7.hs, interpreted )
Ok, one module loaded.
>>> 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]
>>>
```

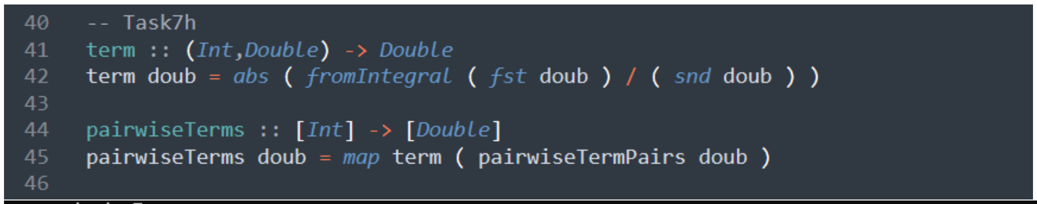
Task 7g:



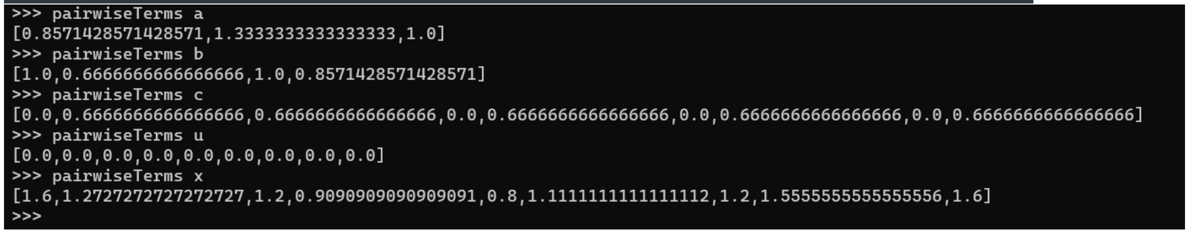
```haskell
-- Task7g
pairwiseTermPairs :: [Int] -> [(Int,Double)]
pairwiseTermPairs num = zip ( pairwiseDifferences num ) ( pairwiseHalfSums num )
```

```
>>> 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)]
>>>
```

Task 7h:

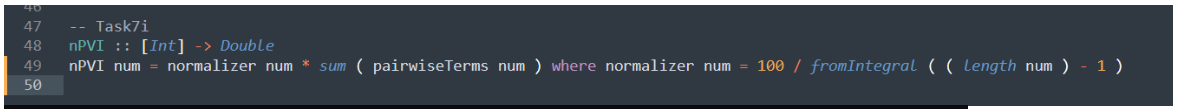


```haskell
-- Task7h
term :: (Int,Double) -> Double
term doub = abs ( fromIntegral ( fst doub ) / ( snd doub ) )

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

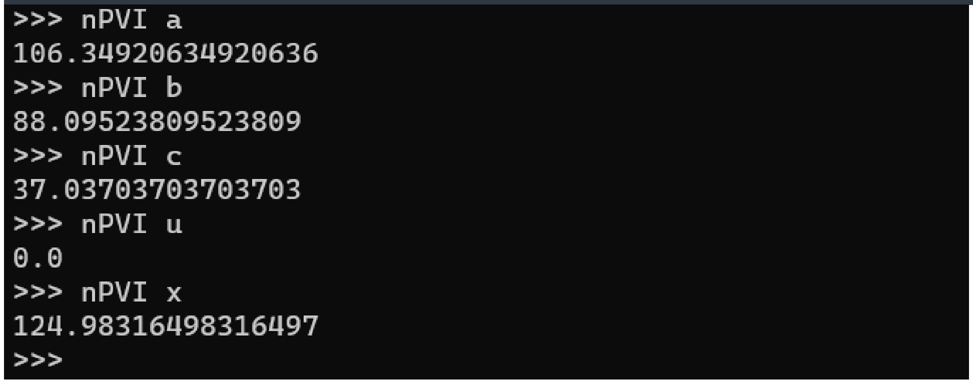


```
>>> pairwiseTerms a
[0.8571428571428571,1.3333333333333333,1.0]
>>> pairwiseTerms b
[1.0,0.6666666666666666,1.0,0.8571428571428571]
>>> pairwiseTerms c
[0.0,0.6666666666666666,0.6666666666666666,0.0,0.6666666666666666,0.0,0.6666666666666666,0.0,0.6666666666666666]
>>> pairwiseTerms u
[0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0]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

Task 7i:



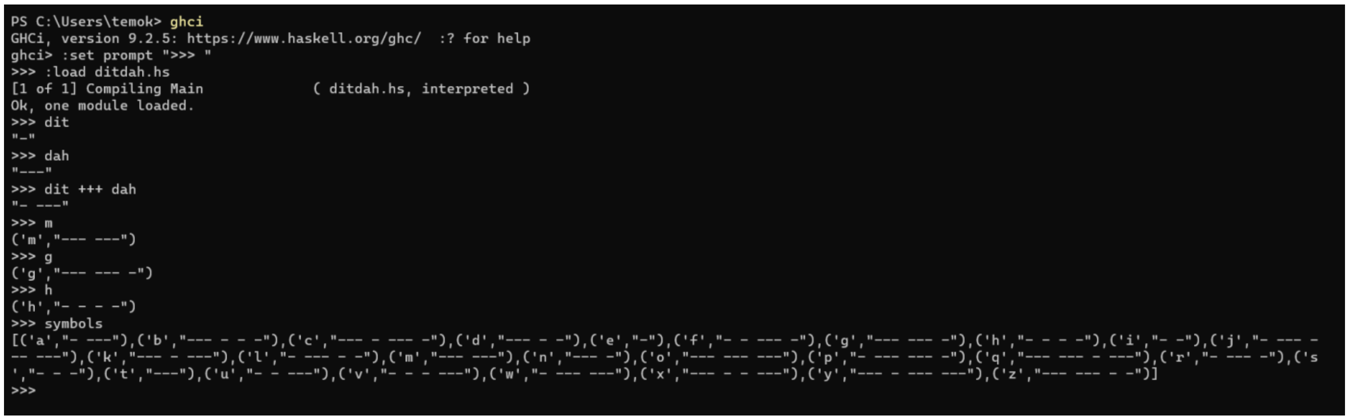
```haskell
-- Task7i
nPVI :: [Int] -> Double
nPVI num = normalizer num * sum ( pairwiseTerms num ) where normalizer num = 100 / fromIntegral ( ( length num ) - 1 )
```



```
>>> nPVI a
106.34920634920636
>>> nPVI b
88.09523809523809
>>> nPVI c
37.03703703703703
>>> nPVI u
0.0
>>> nPVI x
124.98316498316497
>>>
```

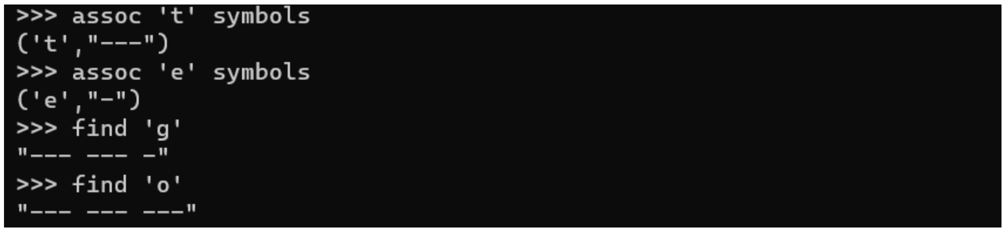
Task 8 - Historic Code: The Dit Dah Code

Task 8a:



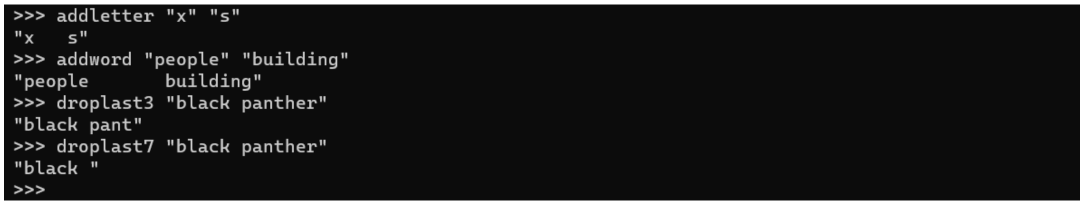
```
PS C:\Users\temok> ghci
GHCi, version 9.2.5: https://www.haskell.org/ghc/  :? for help
ghci> :set prompt ">>> "
>>> :load ditdah.hs
[1 of 1] Compiling Main             ( ditdah.hs, interpreted )
Ok, one module loaded.
>>> dit
"-"
>>> dah
"---"
>>> dit +++ dah
"- ---"
>>> m
('m',"--- ---")
>>> g
('g',"--- --- -")
>>> 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',"--- --- - -")]
>>>
```

Task 8b:



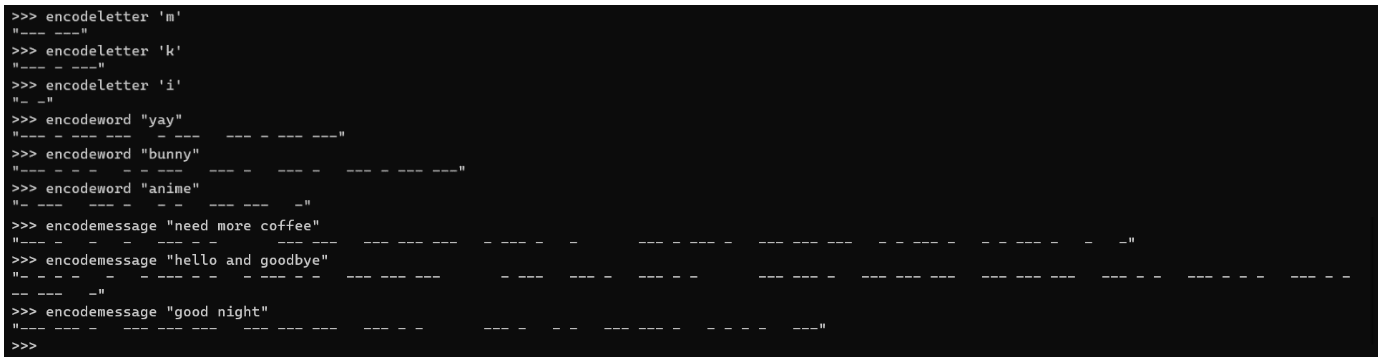
```
>>> assoc 't' symbols
('t',"---")
>>> assoc 'e' symbols
('e',"-")
>>> find 'g'
"--- --- -"
>>> find 'o'
"--- --- ---"
```

Task 8c:



```
>>> addletter "x" "s"
"x   s"
>>> addword "people" "building"
"people       building"
>>> droplast3 "black panther"
"black pant"
>>> droplast7 "black panther"
"black "
>>>
```

Task 8d:



```
>>> encodeletter 'm'
"--- ---"
>>> encodeletter 'k'
"--- - ---"
>>> encodeletter 'i'
"- -"
>>> encodeword "yay"
"--- - --- ---   - ---   --- - --- ---"
>>> encodeword "bunny"
"--- - - -   - - ---   --- -   --- -   --- - --- ---"
>>> encodeword "anime"
"- ---   --- -   - -   --- ---   -"
>>> encodemessage "need more coffee"
"--- -   -   -   --- - -       --- ---   --- --- ---   - --- -   -       --- - --- -   --- --- ---   - - --- -   - - --- -   -   -"
>>> encodemessage "hello and goodbye"
"- - - -   -   - --- - -   - --- - -   --- --- ---       - ---   --- -   --- - -       --- --- -   --- --- ---   --- --- ---   --- - -   --- - - -   --- - -
-- ---   -"
>>> encodemessage "good night"
"--- --- -   --- --- ---   --- --- ---   --- - -       --- -   - -   --- --- -   - - - -   ---"
>>>
```