```
# File: RecursiveFunctions.py
# Student: Jennifer Truong
# UT EID: Jat5244
# Course Name: CS303E
#
# Date Created: 4/26/2021
# Date Last Modified:  4/30/2021
# Description of Program: Multiple functions that uses the recursive method

def sumItemsInList( L ):
    """ Given a list of numbers, return the sum. """

    if L == []:
        return 0
    else:
        return L[0] + sumItemsInList( L[1:] )

def countOccurrencesInList( key, L ):
    """ Return the number of times key occurs in
    list L. """

    if L == []:
        return 0
    elif key == L[0]:
        return 1 + countOccurrencesInList( key, L[1:] )
    else:
        return countOccurrencesInList( key, L[1:] )

def addToN ( n ):
    """ Add up the non-negative integers to n.
    E.g., addToN( 5 ) = 0 + 1 + 2 + 3 + 4 + 5. """

    if n == 0:
       return 0
    else:
        return ( n + addToN( n - 1 ) )

def findSumOfDigits( n ):
    """ Return the sum of the digits in a non-negative integer. """

    if n == 0:
        return 0
    else:
        return n % 10 + findSumOfDigits( int( n / 10 ) )

def decimalToBinary( n ):
    """ Given a nonnegative decimal n, return the
    binary representation as a string. """

    if ( n // 2 ) == 0:
        return str( n % 2 )
    else:
        return (decimalToBinary(n // 2) ) + str(n % 2)


def decimalToList( n ):
    """ Given a nonnegative decimal integer, return a list of the
   digits (as strings). """
```

```
    if n < 10:
        return [ str(n) ]
    else:
        num = []
        num.append(str( n % 10 ))
        return decimalToList ( n // 10) + num


def isPalindrome( s ):
    """ Return True if string s is a palindrome and False
    otherwise. Count the empty string as a palindrome. """

    if len(s) < 2:
        return True
    elif s[0] != s[len(s) -1]:
        return False
    else:
        return isPalindrome(s[1 : len(s) -1])



def findFirstUppercase( s ):
    """ Return the first uppercase letter in
    string s, if any.  Return None if there
    is none. """

    if len(s) == 0:
        return None
    elif s[0].isupper():
        return s[0]
    else:
        return findFirstUppercase( s[1:] )



def findFirstUppercaseIndexHelper( s, index ):
    """ Helper function for findFirstUppercaseIndex. """

    if len(s) == 0:
        return -1
    elif s[0].isupper():
        return index
    else:
        return findFirstUppercaseIndexHelper( s[1 :] , index + 1)



# The following function is already completed for you.  But
# make sure you understand what it's doing.

def findFirstUppercaseIndex( s ):
    """ Return the index of the first uppercase letter in
    string s, if any.  Return -1 if there is none.  This one
    requires a helper function, which is the recursive
    function. """

    return findFirstUppercaseIndexHelper( s, 0 )
```