1. Write a function which reverses a string (e.g. "Don't get sick" becomes "kcis teg t'noD").

```
def reverse( string ):
    if len( string ) < 2:
        return string
    else:
        return reverse( string[1:] ) + string[0]
```

2. Write a function that takes in a string and returns a string that reverses the letters in each word, but keeps the word ordering the same. (e.g. reverse_words("I wear a Stetson now – Stetsons are cool") returns "I raew a nostetS won – snostetS era looc"). You may use the function reverse() from question #**??** in your solution.

```
def reverse_words( string ):
    split_string = string.split()
    return reverse( split_string[0] ) + reverse_words( split_string[1:] )
```

OR

```
def reverse_words( string):
    str = ""
    for word in string.split():
        str += reverse( word )
    return str
```

3. Write a function that takes in a file name, and returns the average size of a word eg. a file containing:

   lots of work
   no rest for midterms
   sad for you

   has an average length of: 3.6

```
def reverse_file( filename ):
    characters = 0
    words = 0
    for line in open(filename):
        for word in line.split():
            words += 1
            characters += len(word)
    return characters/words
```

4. Perform a substitution trace on

```
1    reverse('Cinco-fone')
```

```
1  reverse('Cinco-fone')
2  reverse('inco-fone') + 'C'
3  reverse('nco-fone') + 'i' + 'C'
4  reverse('co-fone') + 'n' + 'i' + 'C'
5  reverse('o-fone') + 'c' + 'n' + 'i' + 'C'
6  reverse('-fone') + 'o' + 'c' + 'n' + 'i' + 'C'
7  reverse('fone') + '-' + 'o' + 'c' + 'n' + 'i' + 'C'
8  reverse('one') + 'f' + '-' + 'o' + 'c' + 'n' + 'i' + 'C'
9  reverse('ne') + 'o' + 'f' + '-' + 'o' + 'c' + 'n' + 'i' + 'C'
10 reverse('e') + 'n' + 'o' + 'f' + '-' + 'o' + 'c' + 'n' + 'i' + 'C'
11 'e' + 'n' + 'o' + 'f' + '-' + 'o' + 'c' + 'n' + 'i' + 'C'
12 'enof-ocniC'
```

5. Write a function that takes in a list of numbers and returns the sum of all of those numbers.

   (a) Recursively.

```
1 def sum( numbers ):
2     if len(numbers) == 0:
3         return 0
4     else
5         return numbers[0] + sum( numbers[1:] )
```

   (b) Iteratively

```
1 def sum( numbers ):
2     index = 0
3     sum = 0
4     while index < len(numbers):
5         sum += numbers[index]
6         index += 1
7     return sum
```

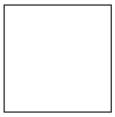
6. How would you test this function?

   - Empty list
   - 1 element list
   - multi-element list

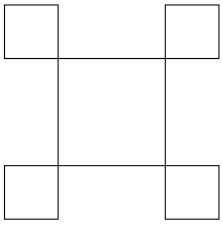
7. Assuming the turtle is facing east, write the python code to draw the following picture given the proper depth as input:

   - depth = 0
     No output

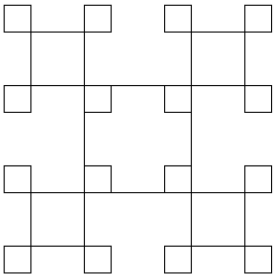
- depth = 1



- depth = 2



- depth = 3



```
1         def drawSqaures( length, depth ):
2             if depth <= 0:
3                 return
4             count = 4
5             while count > 0:
6                 turtle.forward( length )
7                 turtle.left( 90 )
8                 drawSqaures( length/2, depth-1 )
9                 turtle.right( 180 )
10                count -= 1
```

8. What does the following evaluate to?

```
1 def writeThatDown( n ):
2     if n < 5:
```

```
3           return n
4       return (2 * n)
5
6   def he( n ):
7       temp = n + 180
8       if temp > 185:
9           return temp
10      return n
11
12  def putstheFernback( n ):
13      return -n
14
15  n = 20
16  n = he(putstheFernback(writeThatDown( n ) ) )
17  print( n )
```

-40