Sorting and Modules

Sorting

Lists have a `sort` method

Strings are sorted alphabetically, except ...

```
>>> L1 = ["this", "is", "a", "list", "of", "words"]
>>> print L1
['this', 'is', 'a', 'list', 'of', 'words']
>>> L1.sort()
>>> print L1
['a', 'is', 'list', 'of', 'this', 'words']
>>>
```

Uppercase is sorted before lowercase (yes, strange)

```
>>> L1 = ["this", "is", "a", "list", "Of", "Words"]
>>> print L1
['this', 'is', 'a', 'list', 'Of', 'Words']
>>> L1.sort()
>>> print L1
['Of', 'Words', 'a', 'is', 'list', 'this']
>>>
```

```
>>> for i in range(32, 127):
...     print i, "=", chr(i)
...
32 =     56 = 8   80 = P  104 = h
33 = !   57 = 9   81 = Q  105 = i
34 = "   58 = :   82 = R  106 = j
35 = #   59 = ;   83 = S  107 = k
36 = $   60 = <   84 = T  108 = l
37 = %   61 = =   85 = U  109 = m
38 = &   62 = >   86 = V  110 = n
39 = '   63 = ?   87 = W  111 = o
40 = (   64 = @   88 = X  112 = p
41 = )   65 = A   89 = Y  113 = q
42 = *   66 = B   90 = Z  114 = r
43 = +   67 = C   91 = [  115 = s
44 = ,   68 = D   92 = \  116 = t
45 = -   69 = E   93 = ]  117 = u
46 = .   70 = F   94 = ^  118 = v
47 = /   71 = G   95 = _  119 = w
48 = 0   72 = H   96 = `  120 = x
49 = 1   73 = I   97 = a  121 = y
50 = 2   74 = J   98 = b  122 = z
51 = 3   75 = K   99 = c  123 = {
52 = 4   76 = L  100 = d 124 = |
53 = 5   77 = M  101 = e 125 = }
54 = 6   78 = N  102 = f 126 = ~
55 = 7   79 = O  103 = g
```

ASCII order

```
>>> for letter in "Hello":
...     print ord(letter)
...
72
101
108
108
111
10
>>>
```

Sorting Numbers

Numbers are sorted numerically

```
>>> L3 = [5, 2, 7, 8]
>>> L3.sort()
>>> print L3
[2, 5, 7, 8]
>>> L4 = [-7.0, 6, 3.5, -2]
>>> L4.sort()
>>> print L4
[-7.0, -2, 3.5, 6]
>>>
```

Sorting Both

You can sort with both numbers and strings

```
>>> L5 = [1, "two", 9.8, "fem"]
>>> L5.sort()
>>> print L5
[1, 9.8000000000000007, 'fem', 'two']
>>>
```

If you do, it usually means you've designed your program poorly.

Sort returns nothing!

Sort modifies the list “in-place”

```
>>> L1 = "this is a list of words".split()
>>> print L1
['this', 'is', 'a', 'list', 'of', 'words']
>>> x = L1.sort()
>>> print x
None
>>> print L1
['a', 'is', 'list', 'of', 'this', 'words']
>>>
```

Three steps for sorting

#1 - Get the list

```
>>> L1 = "this is a list of words".split()
>>> print L1
['this', 'is', 'a', 'list', 'of', 'words']
```

#2 - Sort it

```
>>> L1.sort()
```

#3 - Use the sorted list

```
>>> print L1
['a', 'is', 'list', 'of', 'this', 'words']
>>>
```

Sorting Dictionaries

Dictionary keys are unsorted

```
>>> D = {"ATA": 6, "TGG": 8, "AAA": 1}
>>> print D
{'AAA': 1, 'TGG': 8, 'ATA': 6}
>>>
```

Sorting Dictionaries

#1 - Get the list

```
>>> D = {"ATA": 6, "TGG": 8, "AAA": 1}
>>> print D
{'AAA': 1, 'TGG': 8, 'ATA': 6}
>>> keys = D.keys()
>>> print keys
['AAA', 'TGG', 'ATA']
>>>
```

#2 - Sort the list

```
>>> D = {"ATA": 6, "TGG": 8, "AAA": 1}
>>> print D
{'AAA': 1, 'TGG': 8, 'ATA': 6}
>>> keys = D.keys()
>>> print keys
['AAA', 'TGG', 'ATA']
>>> keys.sort()
>>> print keys
['AAA', 'ATA', 'TGG']
>>> for k in keys:
...   print k, D[k]
...
AAA 1
ATA 6
TGG 8
>>>
```

#3 - Use the sorted list

```
>>> D = {"ATA": 6, "TGG": 8, "AAA": 1}
>>> print D
{'AAA': 1, 'TGG': 8, 'ATA': 6}
>>> keys = D.keys()
>>> print keys
['AAA', 'TGG', 'ATA']
>>> keys.sort()
>>> print keys
['AAA', 'ATA', 'TGG']
>>> for k in keys:
...     print k, D[k]
...
AAA 1
ATA 6
TGG 8
>>>
```

More info

There is a “how-to” on sorting at
http://www.amk.ca/python/howto/sorting/sorting.html

Modules

Modules are collections of objects (like strings, numbers, functions, lists, and dictionaries)

You've seen the math module

```
>>> import math
>>> math.cos(0)
1.0
>>> math.cos(math.radians(45))
0.7071067811865476
>>> math.sqrt(2) / 2
0.7071067811865476
>>> math.hypot(5, 12)
13.0
>>>
```

Importing a module

The `import` statement tells Python to find module with the given name.

```
>>> import math
>>>
```

This says to import the module named ‘math’.

Using the new module

Objects in the math module are accessed with the "dot notation"

```
>>> import math
>>> math.pi
3.141592653589793
>>>
```

This says to get the variable named "pi" from the math module.

Attributes

The dot notation is used for *attributes*, which are also called *properties*.

```
>>> import math
>>> math.pi
3.141592653589793
>>> math.degrees(math.pi)
180.0
>>>
```

"pi" and "degrees" are attributes (or properties) of the math module.

Make a module

First, create a new file

In IDLE, click on “File” then select “New Window”.
This creates a new window.

In that window, save it to the file name
seq_functions.py

At this point the file is empty.

Add Python code

In the file “seq_functions.py” add the following

```
BASES = "ATCG"

def GC_content(s):
  return (s.count("G") + s.count("C")) / float(len(s))
```

Next, save this file (again).

Test it interactively

```
>>> import seq_functions
>>> seq_functions.BASES
'ATCG'
>>>
seq_functions.GC_content("ATCG")
0.5
>>>
```

Using it from a program

Create a new file called “main.py”

Add the following code

```
import seq_functions
print "%GC content: ", seq_functions.GC_content(seq_functions.BASES)
```

Run this program. You should see 0.5 printed out.

Making changes

If you edit "seq_functions.py" then you must tell Python to reread the statements from the module.

This does not happen automatically.

We have configured IDLE to reread all the modules when Python runs.

If you edit a file in IDLE, you must do "Run Module" for Python to see the changes.

Assignment 30

Make a new program which asks for a DNA sequence as input and prints the GC content as output.

It must use the "seq_functions.py" module to get the GC_content function.

Example output

```
Enter DNA sequence: AATC
%GC content: 25.0
```

Assignment 31

Take the count_bases function from yesterday.
Put it in the "seq_functions.py" module.

Modify your main program so it also prints the number of bases.

```
Enter DNA sequence: AATC
%GC content: 25.0
A: 2
T: 1
C: 1
```

Assignment 32

Start with the program you have to count the number of sequences which have a given property.

From yesterday's exercise, these are individual functions.

Move those functions into the seq_functions.py module. The program output should be unchanged.