LISTS AGAIN

☐ Is it on the midterm

☐ Yes! No!

☐ We learned this stuff (lists) already

☐ We are going into more depth

☐ If I were you, I would learn it

☐ Not everything in chapter 6 will be on the midterm: just stuff we saw in class or the book earlier
LISTS VS STRINGS
FORCED CONVERSIONS

>>> # string
... foo_str = 'foo'
>>> # list
... foo_lst = ['f','o','o']

print foo_str
foo

print foo_lst
['f', 'o', 'o']

print list(foo_str)
['f', 'o', 'o']

print ''.join(foo_lst)
foo

''.join(some_list) -> some_string

list(some_string) -> some_list
>>> foo_str[1]
'o'
>>> foo_lst[1]
'o'
>>> foo_lst[1:]
['o', 'o']
>>> foo_str[1:]
'oo'

some_list[start:stop:step]

JUST LIKE STRINGS
FOR LOOPS (JUST LIKE STRINGS)

```
>>> for a in foo_str:
...     print a,
... foo
```

```
>>> for a in foo_lst:
...     print a,
... foo
```

- LIST OF 1 CHAR STRINGS
- STRING IS SEQUENCE OF 1 CHAR STRINGS
FRUIT LOOPS LISTS VERSION

```python
>>> fruits = ['bannana', 'apple', 'tomato']
>>> for fruit in fruits:
...    print fruit
... 
bannana
apple
tomato
```

LIST OF STRINGS
FRUIT LOOPS STRING VERSION

```python
>>> fruits = 'bannana, apple, tomato'
>>> for fruit in fruits:
...    print fruit
... 
b
a
n
n
a
n
a
,
 
a
p
p
l
e
,
 
t
o
m
a
t
o

ONE CHAR AT A TIME
```
A LIST OF ANYTHING

>>> int_lst = [1,2,3,4,5]
>>> squares = [num**2 for num in int_lst]
>>> print squares
[1, 4, 9, 16, 25]
>>> mixed_list = [1,'frog',1./3,True]
>>> print mixed_list[-2:]
[0.33333333333333331, True]

ENTRIES CAN BE ANYTHING
EVEN CAN HAVE LIST OF LISTS

```python
>>> lists = [ range(1,max_num) for max_num in range(2,5)]
>>> print lists
[[1], [1, 2], [1, 2, 3]]
>>> lists[1][1]
2
```
EVEN LIST OF FUNCTIONS

```python
>>> import math
>>> list_of_f = [math.sin, math.cos, math.sqrt]
>>> vals = [f(math.pi) for f in list_of_f]
>>> print vals
[1.2246467991473532e-16, -1.0, 1.7724538509055159]
```

MUCH MORE GENERAL THAN STRINGS
>>> animal = 'cat'
>>> animal[0] = 'b'
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'str' object does not support item assignment

CAN'T CHANGE A STRING
CONCATENATION RECREATION

>>> animal = 'bob'
>>> animal += 'cat'
>>> print animal
bobcat

ANIMAL ‘BOB’ REPLACED WITH NEW STRING: ‘BOBCAT’

NOT REALLY AN “APPEND”
LISTS REALLY MUTABLE

```python
>>> phrase = ['old','dogs','can','learn','new','tricks']
>>> print phrase
['old', 'dogs', 'can', 'learn', 'new', 'tricks']
>>> phrase[1] = 'cats'
>>> print phrase
['old', 'cats', 'can', 'learn', 'new', 'tricks']
>>>
```

UNLIKE STRINGS
WE CAN CHANGE SINGLE ELEMENTS
ALSO CAN CONCATENATE

>>> phrase = ['old','dogs']+'new','tricks'
>>> print phrase
['old', 'dogs', 'new', 'tricks']

JUST LIKE STRINGS
LIST INFO

len(list_var)
list.count(element)
list.index(element)

>>> fruits = ['apple','bannana', 'apple', 'tomato']
>>> len(fruits)
4
>>> fruits.count('tomato')
1
>>> fruits.count('apple')
2
>>> fruits.index('apple')
0

USEFUL FUNCTIONS
list.append(element)
list.insert(index,element)

```python
>>> things_i_would_like = ['pizza', "a good night's sleep"]
>>> things_i_would_like.append('ipad')
>>> print things_i_would_like
['pizza', 'ipad', "a good night's sleep"]
>>> things_i_would_like.insert(1,'vacation in Rome')
>>> print things_i_would_like
['pizza', 'vacation in Rome', 'ipad', "a good night's sleep"]
```
element = list.pop()
list.remove(element)
del list_var[index]

>>> to_dos = ['feed the meter','water the dog', 'walk the cat']
>>> do_it = to_dos.pop()
>>> print do_it
walk the cat
>>> print to_dos
['feed the meter', 'water the dog']
>>> del to_dos[0]
>>> print to_dos
['water the dog']
>>> letters = list('bannanna')
>>> letters.remove('n')
>>> print letters
['b', 'a', 'n', 'a', 'n', 'n', 'a']
>>> data = [3.8, 9.8, 1.3, -99.6]
>>> print data
[3.7999999999999998,
9.8000000000000007, 1.3,
-99.599999999999994]
>>> data.sort()
>>> print data
[-99.599999999999994, 1.3,
3.7999999999999998, 9.8000000000000007]
>>> data.reverse()
>>> print data
[9.8000000000000007,
3.7999999999999998, 1.3,
-99.599999999999994]
>>> animals = ['pig', 'dog', 'cat', 'fox']
>>> animals.sort()
>>> print animals
['cat', 'dog', 'fox', 'pig']
JOIN (AND SPLIT)

SO JEDI YOU WANT TO BE EH?
UNITE!

building = mortar.join(bricks)

>>> print '|'.join(animals)
cat|dog|fox|pig
>>> animals = ['pig','dog','cat','fox']
>>> print '|'.join(animals)
pig|dog|cat|fox
>>> print ','.join(animals)
pig,dog,cat,fox
>>> print ''.join(animals)
pigdogcatfox
>>> print ' and a '.join(animals)
pig and a dog and a cat and a fox
SEPARATE THEM!

```
>>> animal_string = 'cat|dog|fox|pig'
>>> animal_string.split('|')
['cat', 'dog', 'fox', 'pig']
>>> animal_string = 'cat,dog,fox,pig'
>>> animal_string.split(',

bricks = building.split(mortar)

>>> animal_string = 'cat|dog|fox|pig'
>>> animal_string.split('|')
['cat', 'dog', 'fox', 'pig']
>>> animal_string = 'cat and a dog and 

REALLY
A
STRING METHOD

BRICKS:
LIST ITEMS

MORTAR:
STRING

Monday, October 18, 2010
FUN WITH THE BARD

OLD SCHOOL: THE COMPLETE WORKS OF WILLIAM SHAKESPEARE
def read_text(filename):
    fid = open(filename)
    text = fid.read()
    fid.close()
    return text

READS IT ALL
def strip_punctuation(text):
    text = text.lower()
    chars_used = set(list(text))
    for char in chars_used:
        if not char in string.lowercase:
            text = text.replace(char, ' ')
    return text
def to_word_list(text):
    word_list = text.split()
    return word_list
def count_words(word_list):
    unique_words = list(set(word_list))
    unique_words.sort()
    word_count = []
    num_unique_words = len(unique_words)
    word_count = [word_list.count(word) 
                  for word in unique_words]
    return unique_words, word_count

ISSUE: TAKES FOREVER
HOW CAN WE SEE PROGRESS?
def count_words(word_list):
    unique_words = list(set(word_list))
    unique_words.sort()
    word_count = []
    num_unique_words = len(unique_words)
    print 'Counting words'
    pb = ProgressBar(maxval=num_unique_words).start()
    for ind, word in enumerate(unique_words):
        word_count.append(word_list.count(word))
        pb.update(ind+1)
    print 'word count complete.'
    return unique_words, word_count
def main():
    filename = 'pg100.txt'
    outfile = 'wordcounts.csv'
    complete_works = read_text(filename)
    stripped_text = strip_punctuation(complete_works)
    word_list = to_word_list(stripped_text)
    unique_words, word_count = count_words(word_list)
    print 'Shakespeare used ',len(unique_words), ' distinct words.'
    most_used = max(word_count)
    index = word_count.index(most_used)
    print('The word he used most is "%s"' % unique_words[index])
    fid = open(outfile,'w')
    for word,count in zip(unique_words, word_count):
        fid.write(','.join([word,str(count)])+'
')
    fid.close()
    print 'Done'
THANKS