

More Summary Statistics

March 6, 2012

The Big Picture

Overall Goal

Build a Concordance of a text

- *Locations* of words
- *Frequency* of words

Today: Summary Statistics

- Review material from last class
- Count the number of words in Moby Dick
- Compute the average word length of Moby Dick
- Find the longest word in Moby Dick
- Get the vocabulary size of Moby Dick

Review: Types

- Integers

3

-100

1234

Review: Types

- Integers
- Floats

3

-100

1234

12.7

99.99

1234.0

Review: Types

- Integers
- Floats
- Strings

3

-100

1234

12.7

99.99

1234.0

'12'

'hi'

'Anna!'

Review: Types

- Integers

3

-100

1234

- Floats

12.7

99.99

1234.0

- Strings

'12'

'hi'

'Anna!'

- Booleans

True

False

Review: Statements

- Expression Statements  Gives you output

Review: Statements

- Expression Statements *Gives you output*
- Assignment Statements *Stores a value in a variable*

Review: Statements

- Expression Statements Gives you output
- Assignment Statements *Stores a value in a variable*
- `Print` Statements Prints to the screen

Review: Statements

- Expression Statements Gives you output
- Assignment Statements *Stores a value in a variable*
- `Print` Statements Prints to the screen
- `For` Statements “For each element in `myList`, do something”

Review: Statements

- Expression Statements Gives you output
- Assignment Statements Stores a value in a variable
- `Print` Statements Prints to the screen
- `For` Statements “For each element in `myList`, do something”
- `If` Statements If `A` is true, then do something, otherwise do something else

Review: Other Things

- Lists (a type of **data structure**)

```
[0,1,2]
```

```
['hi','there']
```

```
['hi',0.0]
```

```
[1,2,3,4,5,True,False,'true','one']
```

Review: Other Things

- Lists (a type of **data structure**)

```
[0,1,2]
```

```
['hi','there']
```

```
['hi',0.0]
```

```
[1,2,3,4,5,True,False,'true','one']
```

- Files (an **object** that we can open, read, close)

```
myFile = open(fileName, 'r')
```

The Big Picture

Overall Goal

Build a Concordance of a text

- *Locations* of words
- *Frequency* of words

Today: Summary Statistics

- Review material from last class
- Count the number of words in Moby Dick
- Compute the average word length of Moby Dick
- Find the longest word in Moby Dick
- Get the vocabulary size of Moby Dick

The Big Picture

Overall Goal

Build a Concordance of a text

- *Locations* of words
- *Frequency* of words

Today: Summary Statistics

- Review material from last class
- Count the number of words in Moby Dick
- Compute the average word length of Moby Dick
- Find the longest word in Moby Dick
- Get the vocabulary size of Moby Dick

Do Task 1

The Big Picture

Overall Goal

Build a Concordance of a text

- *Locations* of words
- *Frequency* of words

Today: Summary Statistics

- Review material from last class
- Count the number of words in Moby Dick
- Compute the average word length of Moby Dick
- Find the longest word in Moby Dick
- Get the vocabulary size of Moby Dick

Do Task 1

Python Functions

Preloaded Functions		
<code>len</code>	List OR String	Integer
<code>float</code>	Number (as an Integer, Float, or String)	Float
<code>int</code>	Number (as an Integer, Float, or String)	Integer
<code>str</code>	Integer, Float, String, or List	String
<code>range</code>	Two Integers 1. Start Index (Inclusive) 2. End Index (Exclusive)	List of Integers

These functions *cast* a variable of one type to another type

The Big Picture

Overall Goal

Build a Concordance of a text

- *Locations* of words
- *Frequency* of words

Today: Summary Statistics

- Review material from last class
- Count the number of words in Moby Dick
- Compute the average word length of Moby Dick
- Find the longest word in Moby Dick
- Get the vocabulary size of Moby Dick

Compute the Average Word Length of Moby Dick

```
def avgWordLengthInMobyDick():  
    '''Gets the average word length in MobyDick.txt'''  
  
    return avg
```

Compute the Average Word Length of Moby Dick

```
def avgWordLengthInMobyDick():
    '''Gets the average word length in MobyDick.txt'''
    myList = readMobyDick()
    s = 0
    for word in myList:
        s = s + len(word)
    avg = s/float(len(myList))
    return avg
```

Is our Program Accurate?

```
>>> MDList = readMobyDick()
>>> MDList[0:99]
['CHAPTER', '1', 'Loomings', 'Call', 'me', 'Ishmael.',
'Some', 'years', 'ago--never', 'mind', 'how', 'long',
'precisely--', 'having', 'little', 'or', 'no', 'money', 'in',
'my', 'purse,', 'and', 'nothing', 'particular', 'to',
'interest', 'me', 'on', 'shore,', 'I', 'thought', 'I',
'would', 'sail', 'about', 'a', 'little', 'and', 'see', 'the',
'watery', 'part', 'of', 'the', 'world.', 'It', 'is', 'a',
'way', 'I', 'have', 'of', 'driving', 'off', 'the', 'spleen',
'and', 'regulating', 'the', 'circulation.', 'Whenever', 'I',
'find', 'myself', 'growing', 'grim', 'about', 'the',
'mouth;', 'whenever', 'it', 'is', 'a', 'damp,', 'drizzly',
'November', 'in', 'my', 'soul;', 'whenever', 'I', 'find',
'myself', 'involuntarily', 'pausing', 'before', 'coffin',
'warehouses,', 'and', 'bringing', 'up', 'the', 'rear', 'of',
'every', 'funeral', 'I', 'meet;', 'and']
```

The Big Picture

Overall Goal

Build a Concordance of a text

- *Locations* of words
- *Frequency* of words

Today: Summary Statistics

- Review material from last class
- Count the number of words in Moby Dick
- Compute the average word length of Moby Dick
- Find the longest word in Moby Dick
- Get the vocabulary size of Moby Dick

Get the Longest Word in Moby Dick

```
def getLongestWordInMobyDick():  
    '''Returns the longest word in MobyDick.txt'''  
  
    return longestword
```

Get the Longest Word in Moby Dick

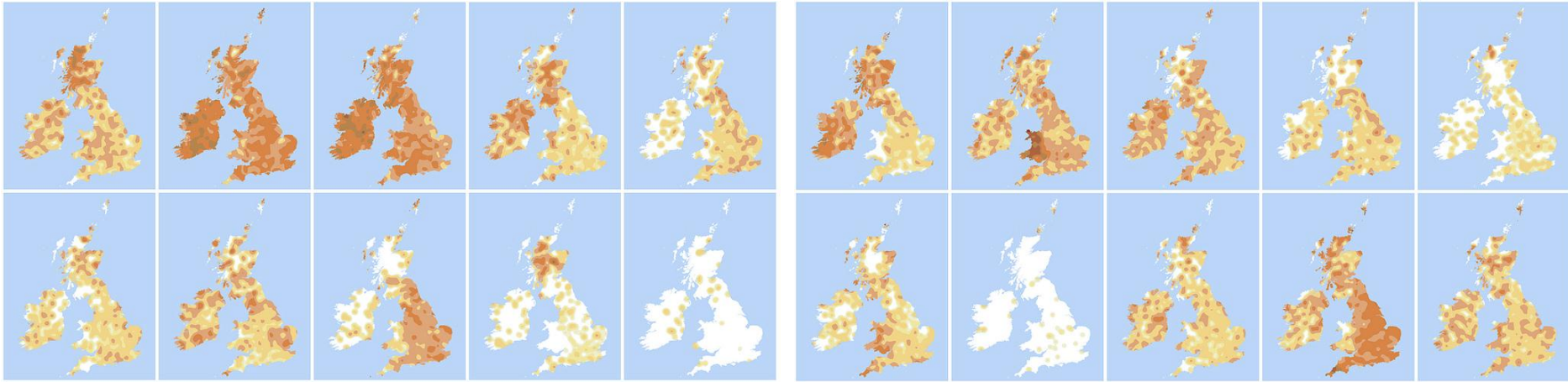
```
def getLongestWordInMobyDick():  
    '''Returns the longest word in MobyDick.txt'''  
    myList = readMobyDick()  
    longestword = ""  
    for word in myList:  
        if len(word) > len(longestword):  
            longestword = word  
    return longestword
```

Get the Longest Word in Moby Dick

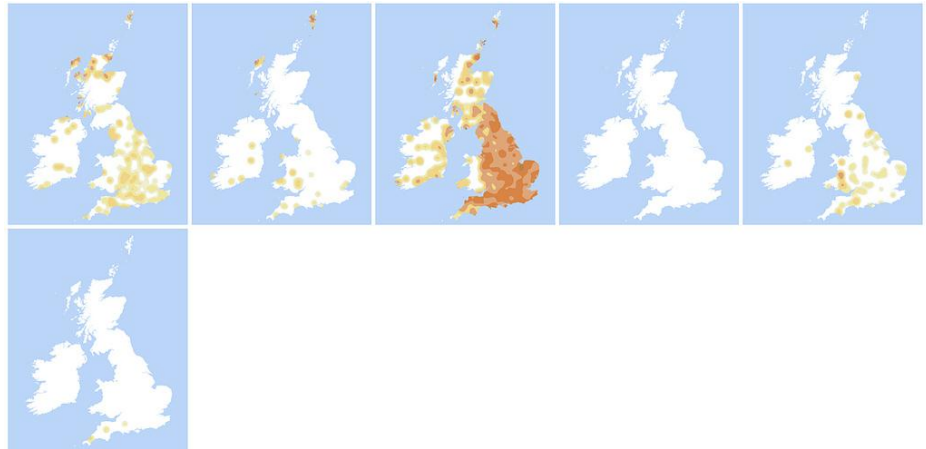
```
def getLongestWordInMobyDick():  
    '''Returns the longest word in MobyDick.txt'''  
    myList = readMobyDick()  
    longestword = ""  
    for word in myList:  
        if len(word) > len(longestword):  
            longestword = word  
    return longestword
```

Is our program accurate?

Break



Alphabet Maps of the UK
and Ireland ([url on site](#))



The Big Picture

Overall Goal

Build a Concordance of a text

- *Locations* of words
- *Frequency* of words

Today: Summary Statistics

- Review material from last class
- Count the number of words in Moby Dick
- Compute the average word length of Moby Dick
- Find the longest word in Moby Dick
- Get the vocabulary size of Moby Dick



The Big Picture

Overall Goal

Build a Concordance of a text

- *Locations* of words
- *Frequency* of words

Today: Summary Statistics

- Review material from last class
- Count the number of words in Moby Dick
- Compute the average word length of Moby Dick
- Find the longest word in Moby Dick
- Get the vocabulary size of Moby Dick

Do Task 2

UNCOMMENT LINES
FROM TASK 1!

Boolean Expressions on Strings

Boolean Operators on Strings		
Operator	Example	Result
Equality	'a' == 'b'	False
Inequality	'a' != 'b'	True
Less Than	'a' < 'b'	True
Less Than or Equal To	'a' <= 'b'	True
Greater Than	'a' > 'b'	False
Greater Than or Equal To	'a' >= 'b'	False

Writing a vocabSize Function

```
def vocabSize():  
    myList = readMobyDick()  
    uniqueList = noReplicates()  
    return len(uniqueList)
```

Writing a vocabSize Function

```
def vocabSize():  
    myList = readMobyDick()  
    uniqueList = noReplicates()  
    return len(uniqueList)
```

```
def noReplicates(wordList):  
    '''takes a list as  
    argument, returns a list free  
    of replicate items.  slow  
    implementation.'''
```

```
def isElementOf(myElement,myList):  
    '''takes a string and a list  
    and returns True if the string is  
    in the list and False  
    otherwise.'''
```

What does `slow` `implementation` mean?

- Re-comment the lines and run `vocabSize()`
 - Hint: Ctrl-C (or Command-C) will abort the call.

What does `slow` `implementation` mean?

- Re-comment the lines and run `vocabSize()`
 - Hint: Ctrl-C (or Command-C) will abort the call.
- There's a *faster* way to write `noReplicates()`
 - What if we can sort the list?
[`'a'`, `'a'`, `'a'`, `'at'`, `'and'`, `'and'`, ..., `'zebra'`]

Python `For` Statements (For Loops)

“For each element in list `myList`, do something”

```
>>> myList = [1,2,3]
>>> for i in range(0,3):
    print myList[i]
```

```
1
```

```
2
```

```
3
```

```
>>>
```

Python `FOR` Statements (For Loops)

“For each element in list `myList`, do something”

```
>>> myList = [1,2,3]
>>> for i in range(0,3):
    print myList[i]
```

1

2

3

>>>

Preloaded Functions

`range`

Two Integers

1. Start Index (Inclusive)

2. End Index (Exclusive)

List of
Integers

Python For Statements (For Loops)

“For each element in list myList, do something”

```
>>> myList = [1,2,3]
>>> for i in range(0,3):
    print myList[i]
```

1
2
3
>>>

List
[0,1,2]

Preloaded Functions		
range	Two Integers 1. Start Index (Inclusive) 2. End Index (Exclusive)	List of Integers

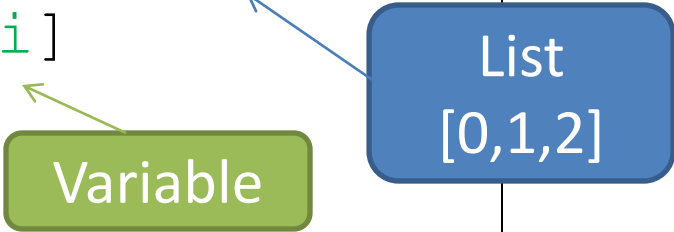
The diagram illustrates the execution of a Python for loop. It shows a code snippet where a list 'myList' is defined as [1, 2, 3] and a for loop iterates over the range(0, 3). A callout box points to the range function, indicating it returns a list of indices [0, 1, 2]. Below the code, a table titled 'Preloaded Functions' provides details for the 'range' function, specifying it takes two integers (start and end) and returns a list of integers.

Python For Statements (For Loops)

“For each element in list myList, do something”

```
>>> myList = [1,2,3]
>>> for i in range(0,3):
    print myList[i]
```

1
2
3
>>>



The diagram consists of two callout boxes. A blue rounded rectangle on the right contains the text 'List [0,1,2]'. A blue arrow points from this box to the 'range(0,3)' part of the code. A green rounded rectangle below it contains the text 'Variable'. A green arrow points from this box to the variable 'i' in the code.

Python For Statements (For Loops)

“For each element in list myList, do something”

```
>>> myList = [1,2,3]
>>> for i in range(0,3):
    print myList[i]
```

1
2
3
>>>

The diagram illustrates the execution of a Python for loop. It shows a code block with three lines of code: `myList = [1,2,3]`, `for i in range(0,3):`, and `print myList[i]`. The code is followed by line numbers 1, 2, and 3, and a prompt `>>>`. A blue rounded rectangle labeled "List [0,1,2]" has a blue arrow pointing to the `range(0,3)` expression in the second line of code. A green rounded rectangle labeled "Variable" has a green arrow pointing to the variable `i` in the same line. A white rounded rectangle with a black border contains the question "Q: What if we don't know the length of the list?".

List [0,1,2]

Variable

Q: What if we don't know the length of the list?

Python For Statements (For Loops)

“For each element in list myList, do something”

```
>>> myList = [1,2,3]
>>> for i in range(0, len(myList)):
    print myList[i]
```

The diagram illustrates the code execution. A green box labeled 'Variable' points to the variable `i` in the `for` loop. A blue box labeled 'List [0,1,2]' points to the `len(myList)` expression in the `range` function.

1
2
3
>>>

Q: What if we don't know the length of the list?

Python `FOR` Statements (For Loops)

“For each element in list myList, do something”

```
>>> def printList(list):  
    for i in range(0, len(list)):  
        print list[i]  
    return
```

Preloaded Functions

range

Two Integers

1. Start Index (Inclusive)
2. End Index (Exclusive)

List of
Integers

Python For Statements (For Loops)

“For each element in list myList, do something”

```
>>> def printList(list):  
    for i in range(0, len(list)):  
        print list[i]  
    return
```

Indentation
Matters!!

Variable

List
[0,1,...len(list)-1]

Returns
NOTHING!