Vocabulary Size of Moby Dick

March 8, 2012

The Big Picture

Overall Goal

Build a Concordance of a text

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

Today: Summary Statistics

- Get the vocabulary size of Moby Dick (Attempt 1)
  - Write test cases to make sure our program works
- Think of a *faster* way to compute the vocabulary size

Boolean Expressions on Numbers

Expression	Evaluation	Final
`(100 > 101) and (-1 != -1)`		
`(1 <= 2) or ((1 == 1) and (1 != 2)`		
`x = 1` `not(x > 2) or (x <= x+1)`		
`y = 100` `not(not(y == 100))`		

Boolean Expressions on Numbers

Expression	Evaluation	Final
`(100 > 101) and (-1 != -1)`	`False and False`	`False`
`(1 <= 2) or ((1 == 1) and (1 != 2)`		
`x = 1` `not(x > 2) or (x <= x+1)`		
`y = 100` `not(not(y == 100))`		

Boolean Expressions on Numbers

Expression	Evaluation	Final
`(100 > 101) and (-1 != -1)`	`False and False`	`False`
`(1 <= 2) or ((1 == 1) and (1 != 2)`	`True or (True and True)`	`True`
`x = 1` `not(x > 2) or (x <= x+1)`		
`y = 100` `not(not(y == 100))`		

Boolean Expressions on Numbers

Expression	Evaluation	Final
`(100 > 101) and (-1 != -1)`	`False and False`	`False`
`(1 <= 2) or ((1 == 1) and (1 != 2)`	`True or (True and True)`	`True`
`x = 1` `not(x > 2) or (x <= x+1)`	`True or True`	`True`
`y = 100` `not(not(y == 100))`		

Boolean Expressions on Numbers

Expression	Evaluation	Final
`(100 > 101) and (-1 != -1)`	`False and False`	`False`
`(1 <= 2) or ((1 == 1) and (1 != 2)`	`True or (True and True)`	`True`
`x = 1` `not(x > 2) or (x <= x+1)`	`True or True`	`True`
`y = 100` `not(not(y == 100))`	`not(False)`	`True`

Boolean Expressions on Strings

Boolean Operators on Strings		
Operator	**Example**	**Result**
Equality	`'a' == 'b'`	`False`
Inequality	`'a' != 'b'`	`True`

Boolean Expressions on Strings

Boolean Operators on Strings		
Operator	**Example**	**Result**
Equality	`'a' == 'b'`	`False`
Inequality	`'a' != 'b'`	`True`

```
>>> 'apple' == 'apple'
True
>>> 'apple' == 'Apple'
False
>>> 'apple' == 'apple!'
False
```

The Big Picture

Overall Goal

Build a Concordance of a text

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

Today: Summary Statistics

- Get the vocabulary size of Moby Dick (Attempt 1)
  - Write test cases to make sure our program works
- Think of a *faster* way to compute the vocabulary size

Save ACT2-4.py and MobyDick.txt to the *same* directory

Writing a `vocabSize` Function

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

Writing a `vocabSize` Function

```
def vocabSize():
    myList = readMobyDickShort()
    uniqueList = noReplicates(myList)
    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.'''
```

Writing a `vocabSize` Function

```
def vocabSize():
    myList = readMobyDickShort()
    uniqueList = noReplicates(myList)
    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.'''
```

```
def testNoReplicates():
```

```
def testIsElementOf():
```

Writing *test cases* are important to make sure your program works!

Break

- Look at *Fusion Tables* (url on website)
  - Created by **Jayant Madhavan** at Google Research

The Big Picture

Overall Goal

Build a Concordance of a text

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

Today: Summary Statistics

- Get the vocabulary size of Moby Dick (Attempt 1)
  - Write test cases to make sure our program works
- Think of a *faster* way to compute the vocabulary size

What does `slow implementation` mean?

- Replace `readMobyDickShort()` with `readMobyDickAll()`
- Now, run `vocabSize()`
  - Hint: Ctrl-C (or Command-C) will abort the call.

What does `slow implementation` mean?

- Replace `readMobyDickShort()` with `readMobyDickAll()`
- Now, run `vocabSize()`
  - Hint: Ctrl-C (or Command-C) will abort the call.

- *Faster* way to write `noReplicates()`
  - What if we can sort the list?

```
['a','a','a','at','and','and',…,'zebra']
```

Sorting Lists

Preloaded Functions			
Name	**Inputs**	**Outputs**	**CHANGES**
`sort`	List		Original List!

Sorting Lists

Preloaded Functions			
Name	**Inputs**	**Outputs**	**CHANGES**
`sort`	List		Original List!

```
>>> myList = [0,4,1,5,-1,6]
>>> myList.sort()
>>> myList
[-1, 0, 1, 4, 5, 6]
```

Sorting Lists

Preloaded Functions			
Name	**Inputs**	**Outputs**	**CHANGES**
`sort`	List		Original List!

```
>>> myList = [0,4,1,5,-1,6]
>>> myList.sort()
>>> myList
[-1, 0, 1, 4, 5, 6]
>>> myList = ['b','d','c','a','z','i']
>>> myList.sort()
>>> myList
['a', 'b', 'c', 'd', 'i', 'z']
```

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)

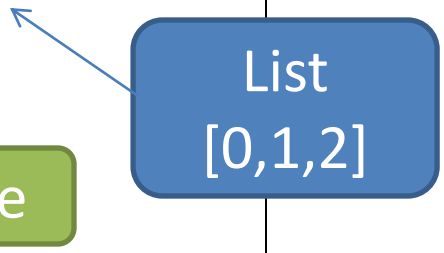
"For each element in list myList, do something"

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


1
2
3
>>>
```

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
>>>
```

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]


1
2
3
>>>
```

List [0,1,2]

Variable

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

The Big Picture

Overall Goal

Build a Concordance of a text

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

Today: Summary Statistics

- Get the vocabulary size of Moby Dick (Attempt 1)
  - Write test cases to make sure our program works
- Think of a *faster* way to compute the vocabulary size
- Practice **debugging** functions

Debugging Programs

Download functionErrors.py & open it in IDLE. Each function has an error or takes an input that results in an error.

Function	Example
`addTwo(x,y)`	`addTwo(1,2)` – output is wrong
`subtractTwo(x,y)`	any inputs cause an error
`multiplyTwo(x,y)`	`z = multiplyTwo(x,y)` – what is z after?
`divideTwo(x,y)`	`divideTwo(2,0)` – write an if statement to "catch" this problem and print a message to the screen.
`addList(myList)`	any inputs cause an error