Project 2: Text Analysis with Python

Header Comments

Mar 14, 2016

Python Dictionaries

Function (All operate on Dictionaries)	Input	Output	Example
`keys()`	None	List of keys	`>>> freq2.keys()` `['the', 'cat']`
`values()`	None	List of values	`>>> freq2.values()` `[3, 2]`
`<key> in <dict>`	Key	Boolean	`>>> 'zebra' in freq2` `False`
`<key> in <dict>`	(same as above)		`>>> 'cat' in freq2` `True`
`del(<dict>[<key>])`	Dict. Entry	None	`>>> del(freq2['cat'])`

- **Keys Are Unique!**
- Assigning/getting any value is very fast

Python Dictionaries

```
The cat had a hat. The cat sat on the hat.
```

I want to write a `wordFreq` function

- What is the input to `wordFreq`?
- What is the output of `wordFreq`?

Word	Freq.
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

Building a Concordance

```
The cat had a hat. The cat sat on the hat.
 0   1   2  3  4    5   6   7   8  9   10
```

Word	List of Positions	Frequency
the	[0,5,9]	3
cat	[1,6]	2
had	[2]	1
a	[3]	1
hat	[4,10]	2
sat	[7]	1
on	[8]	1

The Big Picture

Overall Goal

Build a Concordance of a text

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

Today: Get Word Frequencies

- Define the inputs and the outputs
- Learn a new *data structure*
- Write a function to get word frequencies
- Go from word frequencies to a concordance (finally!)

This will be part of your next HW(2- 7)

Long timeline…

Sun	Mon	Tues	Wed	Thurs	Fri	Sat
3/13	3/14	3/15	3/16	3/17	3/18	3/19
		HW 2-5 due **Project 2: Proposal out**		**HW 2-6 due**		
3/20	3/21	3/22	3/23	3/24	3/25	3/26
		HW 2-7 due	**Initial Proposal due**		**HW 2-8 due**	
3/27	3/28	3/29	3/30	3/31	4/1	4/2
	Revised Proposal due					
4/3	4/4	4/5	4/6	4/7	4/8	4/9
				Project due		

Today's first topic: Project 2

- Reminders
- Data Sources
  - Project Gutenberg
  - English Dictionary
  - Debate Transcripts
- Project 2 Description
- Example Project 2 Proposal

Data Sources

- Looking at a few examples today

Data Sources

- Looking at a few examples today
- Think about what hypotheses you could explore using these data sources

Data Sources

- Looking at a few examples today
- Think about what hypotheses you could explore using these data sources
- What other sources are you interested in?
  - What are the important data you want to compute by extracting pieces of the text?

Data Sources

- Open “Text Data Sources” link on the webpage

http://www.gutenberg.org/

Project Gutenberg

Project Gutenberg

search book catalog

- Book Search
- Catalog
- Bookshelves

search site

- Main Page
- Categories
- News
- Contact Info

donate

Project Gutenberg needs your donation!

Free eBooks by Project Gutenberg

From Project Gutenberg, the first producer of free ebooks.

Mobile Site · Book search · Bookshelves by topic · Top downloads · Recently added · Report errors

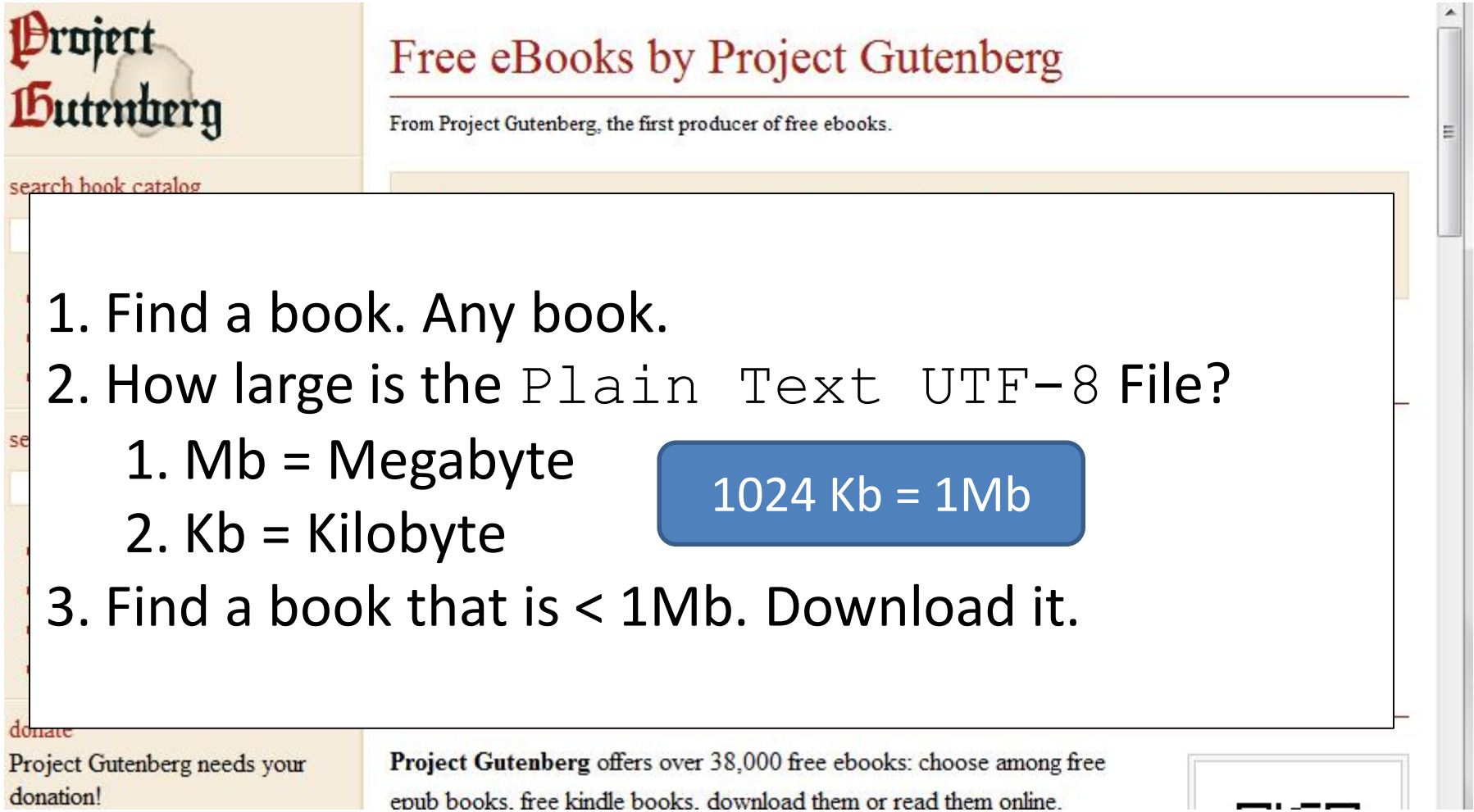
Some of Our Latest Books

Welcome

Project Gutenberg offers over 38,000 free ebooks: choose among free epub books, free kindle books, download them or read them online.

http://www.gutenberg.org/

Project Gutenberg



1. Find a book. Any book.
2. How large is the `Plain Text UTF-8` File?
    1. Mb = Megabyte
    2. Kb = Kilobyte
3. Find a book that is < 1Mb. Download it.

1024 Kb = 1Mb

http://www.gutenberg.org/

Project Gutenberg

Look at the function
`removeLicenseFromProjectGutenberg`
in `DataImport.py`

Today's first topic: Project 2

- Data Sources
  - Project Gutenberg
  - English Dictionary
  - Debate Transcripts
- Project 2 Description
- Example Project 2 Proposal

http://www.mso.anu.edu.au/~ralph/OPTED/

Webster's Unabridged Dictionary

The Online Plain Text English Dictionary

OPTED is a public domain English word list dictionary, based on the public domain portion of "The Project Gutenberg Etext of Webster's Unabridged Dictionary" which is in turn based on the 1913 US Webster's Unabridged Dictionary. (See Project Gutenburg)

This version has been extensively stripped down and set out as one definition per line. All the Gutenburg EText tags and formatting have been removed by computer. Version 0.03 is a new processing of v0.47 of the websters dictionary and it has considerably fewer errors. Also the definition limit of 255 chars has been removed to give full justice of some of the more majestic of the originals. Some important errors in the parts-of-speech fields have been corrected and a lot of inflections/ alternatives and plurals that were missed due to software bugs in v0.01 and 0.02 are now included properly.

The dictionary is set as a word list with definitions, using minimal HTML markup. The only tags used are <P>, <B> and <I> and these serve to delimit the words (between <B>s) the part of speech or type (between <I>s) and the definitions (The rest of the line). Each entry is between a <P>, </P> pair. This will facilitate computer processing. The text was prepared on a macintosh, so the few accented and umlauted characters appear best if your browser is set to Western MacRoman encoding (this should look like an umlauted u : **ü**). If this causes problems and I get enough responses, I'll look into producing an ISO 8859-1 or even a Unicode version.

The dictionary can be viewed (with patience) directly online as you would a normal printed dictionary, otherwise a user can download the pages and process them in some way on their own machine. The only usage conditions are that if the material is redistributed, the content (not the formatting) remain in the public domain (ie free) and that the content be easily accessible in non-encoded plain text format at no cost to the end user. The origin of the content should also be acknowledged, including OPTED, Project Gutenburg and the 1913 edition of Webster's Unabridged Dictionary. If the material is to be included in commercial products, Project Gutenburg should be contacted first. There are no restrictions for personal or research uses of this material.

OPTED v0.03 by Letter(size)

Second computer generated version:

A(1.1M) | B(1005k) | C(1.6M) | D(1M) | E(809k) | F(784k) | G(564k) | H(686k) | I(833k) | J(172k) | K(172k) | L(637k) | M(931k) | N(343k) | O(466k) | P(1.5M) | Q(147k) | R(931k) | S(2.1M) | T(1005k) | U(343k) | V(343k) | W(490k) | X(49k) | Y(74k) | Z(74k)

http://www.mso.anu.edu.au/~ralph/OPTED/

Webster's Unabridged Dictionary

The Online Plain Text English Dictionary

OPTED is a public domain English word list dictionary, based on the public domain portion of "The Project Gutenberg Etext of Webster's Unabridged Dictionary" which is in turn based on the 1913 US Webster's Unabridged Dictionary. (See Project Gutenberg)

1. According to the homepage, what does each line contain?
2. What letter is the **smallest** file?
    1. Mb = Megabyte
    2. Kb = Kilobyte

1024 Kb = 1Mb

3. Click on it. Right-click and select `View Page Source...`

Second computer generated version.

A(1.1M) | B(1005k) | C(1.6M) | D(1M) | E(809k) | F(784k) | G(564k) | H(686k) | I(833k) | J(172k) | K(172k) | L(637k) | M(931k) | N(343k) | O(466k) | P(1.5M) | Q(147k) | R(931k) | S(2.1M) | T(1005k) | U(343k) | V(343k) | W(490k) | X(49k) | Y(74k) | Z(74k)

http://www.mso.anu.edu.au/~ralph/OPTED/

Webster's Unabridged Dictionary

The Online Plain Text English Dictionary

OPTED is a public domain English word list dictionary, based on the public domain portion of "The Project Gutenberg Etext of Webster's Unabridged Dictionary" which is in turn based on the 1913 US Webster's Unabridged Dictionary. (See Project Gutenburg)

This version has been extensively stripped down and set out as one definition per line. All the Gutenburg EText tags and formatting have been removed by computer. Version 0.03 is a new processing of v0.47 of the websters dictionary and it has considerably fewer errors. Also the definition limit of 255 chars has been removed to give full justice of some of the more majestic of the originals. Some important errors in the parts-of-speech fields have been corrected and a lot of inflections/ alternatives and plurals that were missed due to software bugs in v0.01 and 0.02 are now included properly

Look at the function
`getWebsterDictionary`
in `DataImport.py`

and process them in some way on their own machine. The only usage conditions are that if the material is redistributed, the content (not the formatting) remain in the public domain (ie free) and that the content be easily accessible in non-encoded plain text format at no cost to the end user. The origin of the content should also be acknowledged, including OPTED, Project Gutenburg and the 1913 edition of Webster's Unabridged Dictionary. If the material is to be included in commercial products, Project Gutenburg should be contacted first. There are no restrictions for personal or research uses of this material.

OPTED v0.03 by Letter(size)

Second computer generated version:

A(1.1M) | B(1005k) | C(1.6M) | D(1M) | E(809k) | F(784k) | G(564k) | H(686k) | I(833k) | J(172k) | K(172k) | L(637k) | M(931k) | N(343k) | O(466k) | P(1.5M) | Q(147k) | R(931k) | S(2.1M) | T(1005k) | U(343k) | V(343k) | W(490k) | X(49k) | Y(74k) | Z(74k)

Today's first topic: Project 2

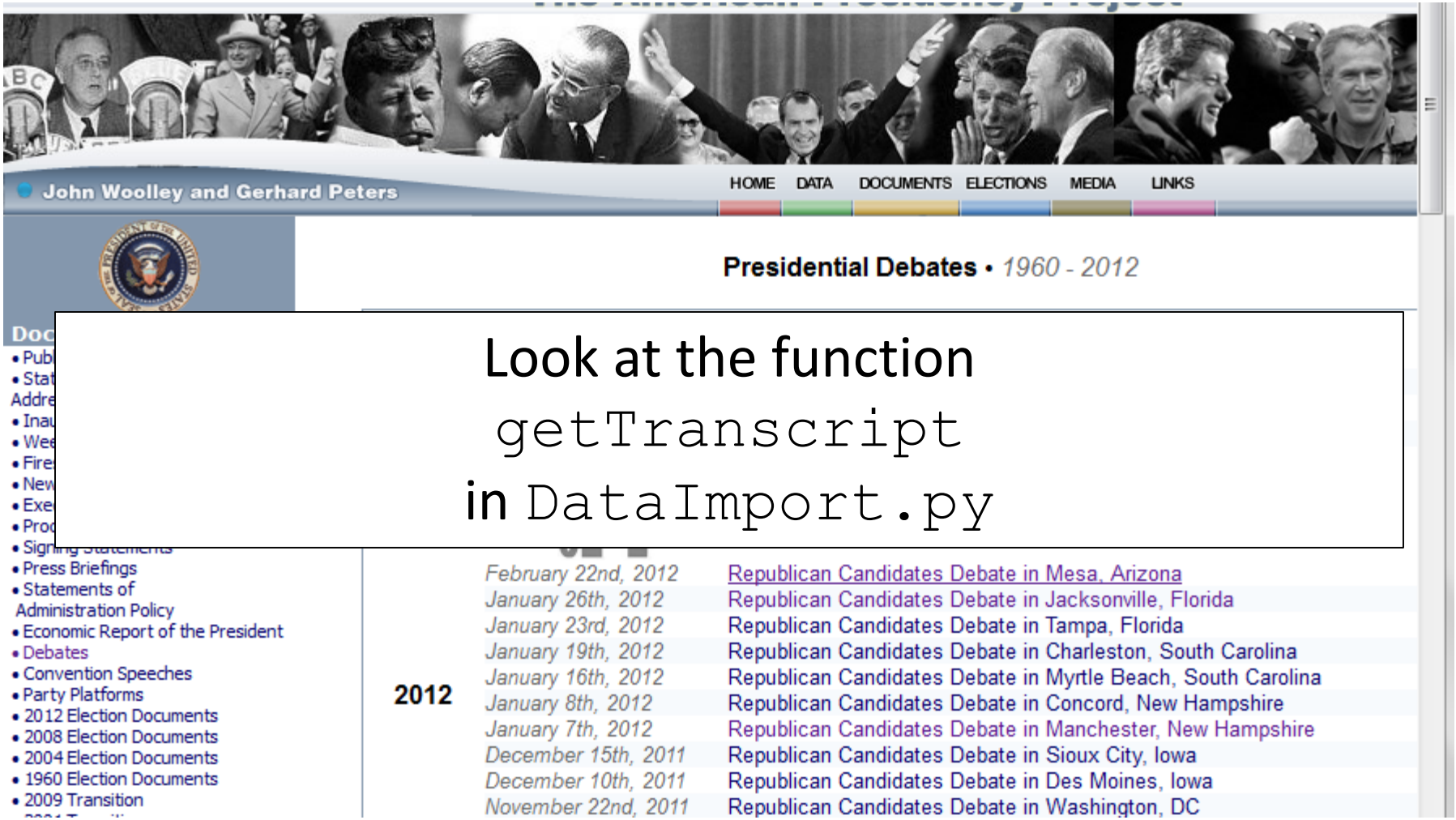
- Data Sources
  - Project Gutenberg
  - English Dictionary
  - Debate Transcripts
- Project 2 Description
- Example Project 2 Proposal

http://www.presidency.ucsb.edu/

The American Presidency Project

http://www.presidency.ucsb.edu/

The American Presidency Project



Look at the function
`getTranscript`
in `DataImport.py`

Today's first topic: Project 2

- Data Sources
  - Project Gutenberg
  - English Dictionary
  - Debate Transcripts
- Project 2 Description
- Example Project 2 Proposal

Project 2 Rubric

Category	# Points Earned
Proposal	25
Design Elements	20
Execution	25
Code Quality	15
Website Presentation and Discussion	15
TOTAL	100

Today's first topic: Project 2

- Data Sources
  - Project Gutenberg
  - English Dictionary
  - Debate Transcripts
- Project 2 Description
- Example Project 2 Proposal

Anna Ritz
Project 2 Proposal

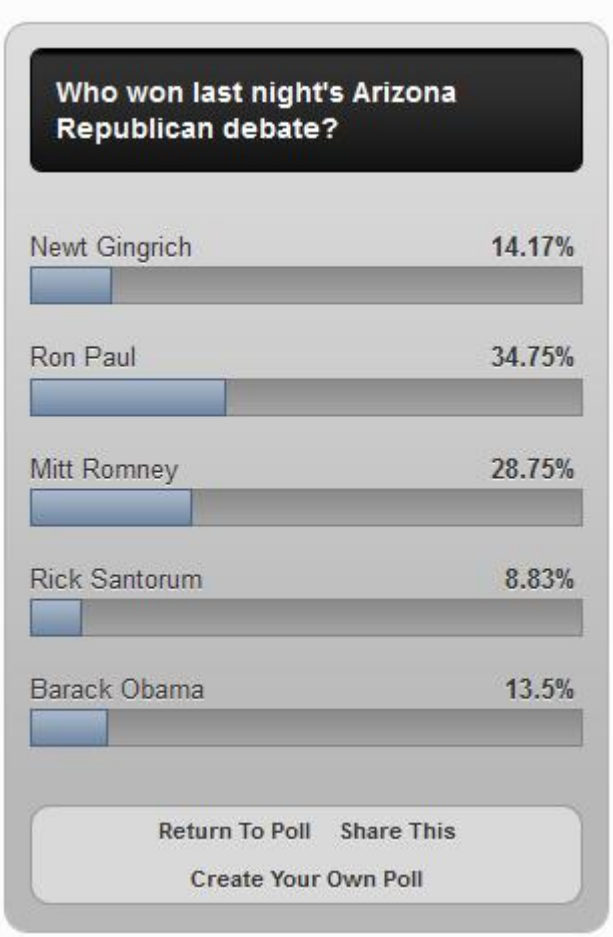
Background: After each debate, there's lots of talk about who "won" it, i.e.

`http://www.washingtonpost.com/blogs/the-fix/post/arizona-republican-debate-winners-and-losers/2012/02/22/gIQAsKkVUR_blog.html`

I will define the "winner" as the person who received applause the most frequently during the debate.

Claim: I claim that in the AZ debate, Romney "won" and Santorum "lost" – that is, Romney received applause the most and Santorum received applause the least.

....

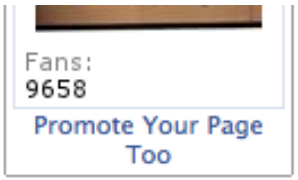


http://blogs.phoenixnewtimes.com/valleyfever/2012/02/who_won_last_nights_arizona_re.php

Look at the file structure…

and it's been broken by this president.

I want to restore America's promise, and I'm going to do that — [*applause*] —That's good enough. As George Costanza would say, when they're applauding, stop. Right?



GINGRICH: I'm Newt Gingrich.

And I've developed a program for American energy so no future president will ever bow to a Saudi king again and so every American can look forward to $2.50 a gallon gasoline. [*applause*]

KING: Gentlemen, it's good to see you again.

Let's get started on the important issues with a question from our audience.

Sir, please tell us who you are and state your question.

UNKNOWN: My name is Gilbert Fidler from Gilbert, Arizona, and I'd like to ask this question to all the candidates if I could.

Since the first time in 65 years our national debt exceeds our gross national product, what are you going to do to bring down the debt?

KING: Thank you, sir.

Senator Santorum, let's begin with you.

SANTORUM: Thank you, Gilbert.

I put together a specific plan that cuts $5 trillion over five years, that spends less money each year for the next four years that I'll be president of the United States. So it's not inflation- adjusted, it's not baseline-budgeting. We're actually going to shrink the actual size of the federal budget, and we're going to do so by dealing with the real problem.

Skeleton Code

Skeleton Code

```
# Anna Ritz
# Project 2
# Skeleton Code

## This program assesses how "popular" each republican candidate is
## by counting the number of [applause] tags in a
## Republican debate.

# CONSTANT VARIABLES (will NEVER change values) are in ALL CAPS
# If you put these variables OUTSIDE all functions, then you
# can access them in ANY function.
CANDIDATES = ['GINGRICH','PAUL','ROMNEY','SANTORUM']
DEBATE_FILE = 'AZDebate.txt'

def assessPopularity():
    '''Assesses the popularity of the candidates in the AZ debate.
    INPUTS: none
    OUTPUTS: none'''

    # Step 1: Read the debate file
    myString = readFile()

    # Step 2: For each candidate, assess popularity
    for cand in CANDIDATES:
        countApplause(cand,myString)

    return

def readFile():
    '''Reads DEBATE_FILE and returns a string.
    INPUTS: none
    OUTPUTS: String of the debate'''

    return '' # returns an empty string for now.

def countApplause(candidate,debateString):
    '''Assesses the popularity of the candidate in the debateString.
    INPUTS: candidate (String) - name of candidate
```

Anna Ritz
Project 2 Proposal

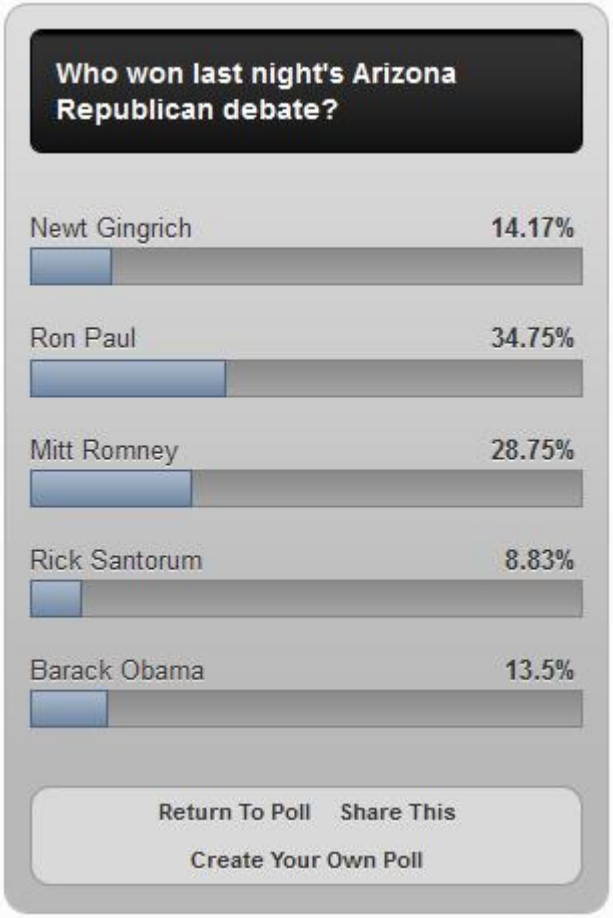
...

Claim: I claim that in the AZ debate, Romney "won" and Santorum "lost" – that is, Romney received applause the most and Santorum received applause the least.

....

Backup Plan: ???

Increasing Degree of Difficulty: ???



http://blogs.phoenixnewtimes.com/valleyfever/2012/02/who_won_last_nights_arizona_re.php

What else can I do?

- Count presence of characters in different chapters in a book.
  - Generate CSV, plot graph on Google Spreadsheets
- Look at the Sherlock Holmes stories
  - Search for “elementary” and “Watson” close together
  - Get all variations of the famous quote (that some people claim it was never said in the book)

What else can I do?

- Get tweets from Western US and Eastern US
  - Check whether “Pepsi” shows up more than “Coke”
  - Soda vs. Pop “issue”

- Right now, we give you tweets in a CSV file
- Later in the course, you’ll get your own tweets

Today's first topic: Project 2

- Data Sources
  - Project Gutenberg
  - English Dictionary
  - Debate Transcripts
- Project 2 Description
- Example Project 2 Proposal

HW: Building a Concordance

```
The cat had a hat. The cat sat on the hat.
0   1   2   3  4    5   6   7   8  9   10
```

Word	List of Positions	Frequency
the	[0,5,9]	3
cat	[1,6]	2
had	[2]	1
a	[3]	1
hat	[4,10]	2
sat	[7]	1
on	[8]	1

List as values in a dictionary

Lists as values of a dictionary

```
The cat had a hat. The cat sat on the hat.
```

Key	Value

```
>>> conc = {}
>>> conc
{}
```

Lists as values of a dictionary

```
The cat had a hat. The cat sat on the hat.
```

Key	Value
cat	[1,6]

```
>>> conc = {}
>>> conc
{}
>>> conc['cat'] = [1,6]
>>> conc
{'cat':[1,6]}
```

Lists as values of a dictionary

```
The cat had a hat. The cat sat on the hat.
```

Key	Value
cat	[1,6]
hat	[4,10]

```
>>> conc = {}
>>> conc
{}
>>> conc['cat'] = [1,6]
>>> conc
{'cat':[1,6]}
>>> conc['hat'] = [4,10]
>>> conc
{'hat':[4,10], 'cat':[1,6]}
```

Lists as values of a dictionary

```
The cat had a hat. The cat sat on the hat.
```

Key	Value
cat	[1,6,400]
hat	[4,10]

```
>>> conc['cat'] = conc['cat'] + [400]
{'cat':[1,6,400], 'hat':[4,10]}
```

Header Comments

Header Comments

```
def addOne(t):
'''Receives a number and returns the number
summed to one'''

def addOne(t):
'''num -> num
Receives a number and returns the number
summed to one'''
```

Header Comments

```
def sumThem(a, b):
'''Receives two integers and returns their
sum'''
```

```
def sumThem(a, b):
'''int * int -> int
Receives two integers and returns their
sum'''
```

Header Comments

```
def buildFreqTable(text):
'''Receives a text and returns a dictionary
mapping each word with its frequency'''

def buildFreqTable(text):
'''string -> (string,int)dict
Receives a text and returns a dictionary
mapping each word with its frequency'''
```

Header Comments

```
def addPassword(dictionary,key,value):
'''Adds the (key,value) pair to the
dictionary and returns the new dictionary'''
```

```
def addPassword(dictionary,key,value):
'''(string,string)dict * string * string ->
(string, string)dict
Adds the (key,value) pair to the dictionary
and returns the new dictionary'''
```

Header Comments

```
def isElementOf(element, listOfElems):
'''Checks if element is part of the provided
list'''

def isElementOf(element, listOfElems):
'''int * int list -> bool
Checks if element is part of the provided
list'''
```

Header Comments

```
def isElementOf(element, listOfElems):
'''Checks if element is part of the provided
list'''
```

```
def isElementOf(element, listOfElems):
'''object * list -> bool
Checks if element is part of the provided
list'''
```

Header Comments

- Notation for describing types:

```
int, float, string, bool
```

- Separate multiple arguments with “*”:

```
open(filename, “r”)

string * string -> file
```

Header Comments

- Also say what the function *produces* in via its return statement:

```
def printMovieRevenues(movie_dict):
'''(string, int) dict -> .
    #some print commands here…
    #some extra stuff particular to the
function…
```

- Use "." to mean "nothing at all"

More complicated types

- Dictionaries

```
(string, int)dict
(string, string list)dict
```

- Lists

```
int list                          [2, 3, 4]
string list                  ['cat', 'zebra']
string list list    [['a', 'b'],['cat', 'h']]
```

- Use parentheses to clarify as needed

```
(string list) list
```

Synonyms

- OK to use “text” for a long string that represents a whole sentence or book, etc.
- OK to use “word” for a string containing an individual word.

```
def getMobyWords(fileString):
'''  text -> string list
     split text of Moby Dick into individual
     words'''
    return fileString.split()
```

Next Classes

- String functions in Python (split, search, etc)
- Get input from the user's keyboard!
- Generate Files
- Using Python to compute a similarity score between books
  - "Which book might have been authored by someone different than the rest?"