Checkpoint 1
Feb 27th 2017

Task 1. Make sure you can get and run!
Download files from the github repo. Make sure you can run them all. Either type "python <filename.py>" for each file / open in IDLE and run each one/ or on linux/mac you may use the test bash script in regTest. If you cannot complete Task 1, see the instructor RIGHT AWAY!

Task 2. Get emails in as Documents!
Look at the sentence.py, document.py, documentReader.py, userException.py. We will be writing the code to read emails from the Enron dataset with the class in documenReader.py and will create a Document object from the document.py file. First, note the information we are storing in our Document object. Most of it is self-explanatory based on variable names or comments. However, self__sentences is odd. Here we will be storing the message (the body of text) in the email. However, we are not just storing the message as a string, but as a list of Sentence objects (sentence.py). Each one being one sentence (A set of characters ending in "." "!" or "?").

Understanding the input data. The data we will be using is that from the Enron email dataset. We will not be using the whole dataset as it is very large and has many errors. (In fact, I will keep adding to the repo more emails as I clean them up for our use.) Therefore, we will only be looking at the dataset provided to us in the **train** and **eval** folder in the repo. Please take some time to look over both the file structure and the email files. In both the train and eval folders there are multiple folders with email addresses. Right now eval folder is empty, since we are not making predictions yet. Emails with in these folders are "From" that sender (as this information is some time hard to parse from the email itself if CCed, replied, forwarded, automated, etc).

The emails will have a structure similar to the one below. Note that emails from replies and forwards may have slightly different structure. In the example, below we see the sender (From) enron_update@concureworkplace.com and receiver (To) rick.buy@enron.com . We will ignore the X-From and X-To. We can also see a date, a subject and after all the header the message "The Following reports have been waiting for your approval for more than 4 days. Please review."

```
Message-ID: <31927470.1075840380493.JavaMail.evans@thyme>
Date: Mon, 23 Jan 0002 21:23:13 -0800 (PST)
From: enron_update@concureworkplace.com
To: rick.buy@enron.com
Subject: Expense Reports Awaiting Your Approval
Mime-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
X-From: enron_update@concureworkplace.com
X-To: Buy, Rick </O=ENRON/OU=NA/CN=RECIPIENTS/CN=RBUY>
X-cc:
X-bcc:
X-Folder: \rbuy\Inbox
X-Origin: BUY-R
X-FileName: richard buy 1-30-02..pst

The following reports have been waiting for your approval for more than 4 days.\
  Please review.
```

Your task is to convert these email files into Document objects! A few precautions, we want to get the "From" from the folder name and not the email file because this can be hard with pseudo-names. If there are multiple "To", only store the first one. Also, we do not want to consider any text below the message, e.g., other headers and messages from forwarded or replied emails. This is a hard job, but great practice at file I/O, using multiple files, and multiple object classes. You may choose to use the exception for fileIO in the userException.py. At this time, these exceptions are optional, but may be required later in the project or used for extra credit. Remember to follow good style and adhere to the Final Project Description