Important

1. Due Date: **Thursday, March 2nd**
2. This homework is graded out of 100 points.
3. This is an Individual Assignment. You may collaborate with other students in this class. Collaboration means talking through problems, assisting with debugging, explaining a concept, etc. Students may only collaborate with fellow students currently taking CS 1301, the TA's and the lecturer. You should not exchange code or write code for others. For individual assignments, each student must turn in a unique program. Your submission must not be substantially similar to another student's submission. Collaboration at a reasonable level will not result in substantially similar code.
4. For Help:
   - TA Helpdesk (Schedule posted on class website.)
   - Email TA's or use Piazza Forums Notes:
   - How to Think Like a Computer Scientists [http://openbookproject.net/thinkcs/python/english3e/]
   - CS 1301 Python Debugging Guide [http://www.cc.gatech.edu/classes/AY2016/cs1301_spring/CS-1301-Debugging-Guide/index.html]
5. Don't forget to include the required collaboration statement (outlined on the syllabus). Failing to include the Collaboration Statement will result in no credit.
6. Do not wait until the last minute to do this assignment in case you run into problems.
7. Comment or delete all your function calls. When your code is run, all it should do is build without any errors. Having function calls or extraneous code outside the scope of functions will result in a lower grade. (import statements and global variables are okay to be outside the scope of a function)
8. **Read the entire specifications document before starting this assignment.**
9. **IF YOUR CODE CANNOT RUN BECAUSE OF AN ERROR, IT IS A 0%.**

Introduction

The goal of this homework is for you to showcase your knowledge about how to properly manipulate files and use some modules to connect to the internet, specifically the `urllib` module. There are different sections for this assignment, for each section you will need to implement some functions. Refer to the rubric to see how points will be rewarded for each section. You have been given `HW6.py` to complete. Some functions have been written out for your usage. Read the docstrings for more information. Also, below you will find more detailed information to complete your assignment. Read it thoroughly before you begin. You have until **Thursday, March 2nd** to complete this assignment. Don't forget to include your name and your collaboration statement. Re-download your submission from T-Square after you submit it to make sure that your code runs successfully.

Part 1: File Manipulation

This section of the assignment provides an introduction to basic file manipulation. After completing these functions, you should have an understanding how how to read, write, and append to text files (.txt).

Function name: `improve_fight_song`
Parameters: `title<str>`
Return value: None

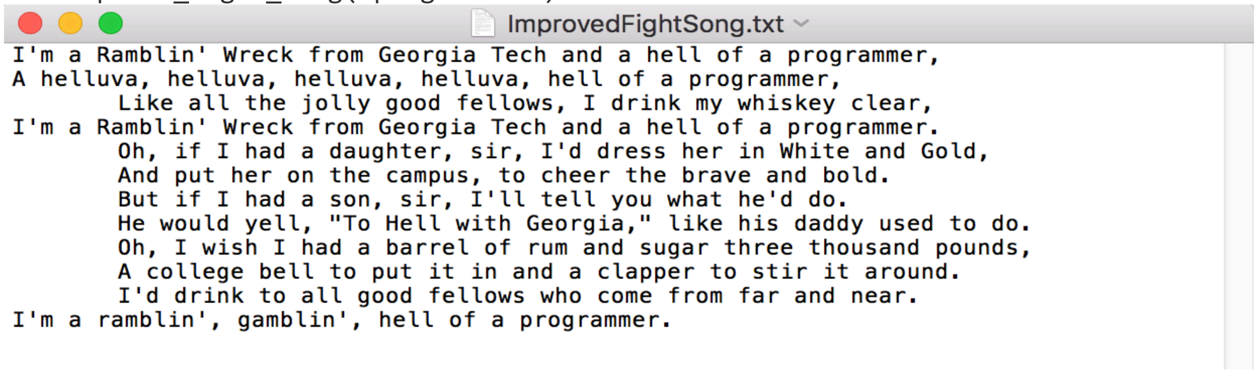
Description: You have been provided with a text file. You can download it here: http://cs1301.com/downloads/RamblinWreck.txt Read the file and replace all instances of 'engineer' with `title<str>`. If `title<str>` begins with a consonant, then the 'an' before the `title<str>` should be replaced with 'a'. Also, indent all lines that do not contain the word 'engineer' (in the original file). The improved fight song is written to a new file. Name this file: **ImprovedFightSong.txt**

Notes:

- The original file RamblinWreck.txt should not be altered.
- If `title<str>` is an empty string, ignore it and only indent the lines that do not contain 'engineer'.

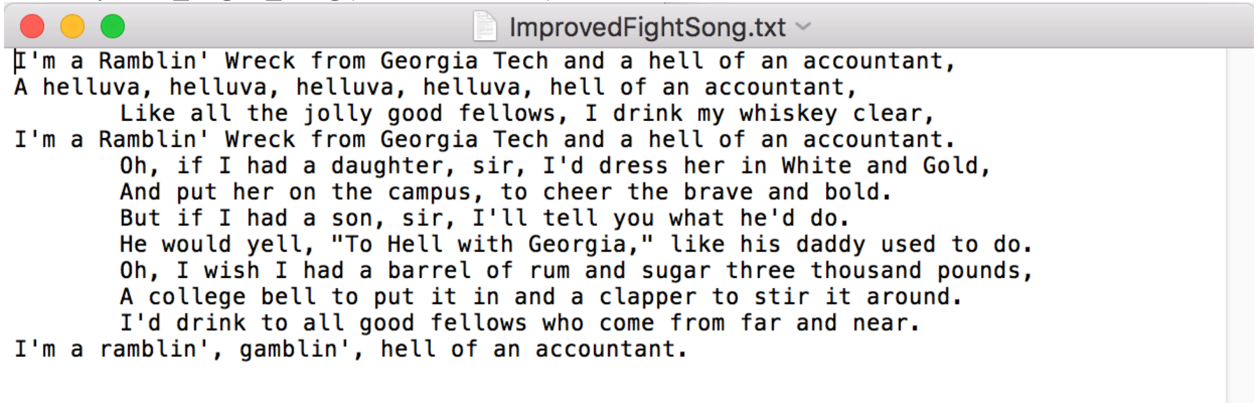
Test Cases:

```
>>> improve_fight_song("programmer")
```



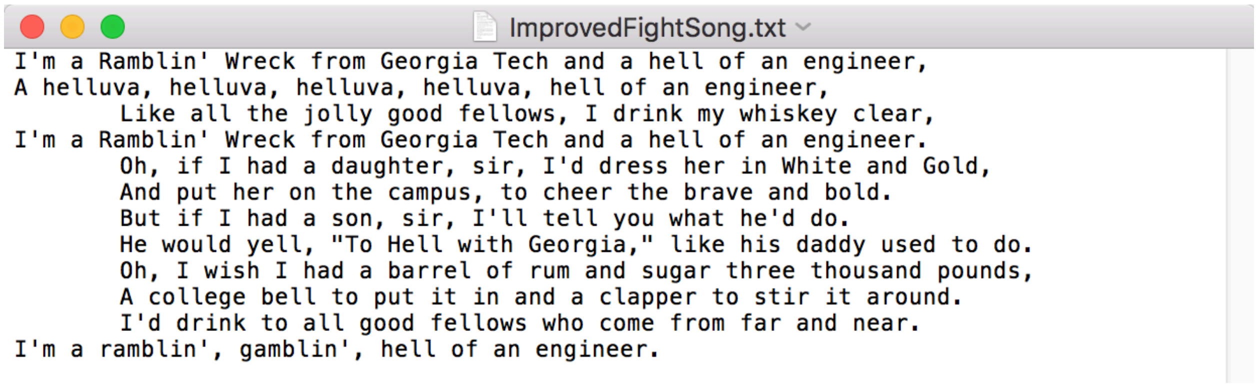
```
I'm a Ramblin' Wreck from Georgia Tech and a hell of a programmer,
A helluva, helluva, helluva, helluva, hell of a programmer,
        Like all the jolly good fellows, I drink my whiskey clear,
I'm a Ramblin' Wreck from Georgia Tech and a hell of a programmer.
        Oh, if I had a daughter, sir, I'd dress her in White and Gold,
        And put her on the campus, to cheer the brave and bold.
        But if I had a son, sir, I'll tell you what he'd do.
        He would yell, "To Hell with Georgia," like his daddy used to do.
        Oh, I wish I had a barrel of rum and sugar three thousand pounds,
        A college bell to put it in and a clapper to stir it around.
        I'd drink to all good fellows who come from far and near.
I'm a ramblin', gamblin', hell of a programmer.
```

```
>>> improve_fight_song("accountant")
```



```
I'm a Ramblin' Wreck from Georgia Tech and a hell of an accountant,
A helluva, helluva, helluva, helluva, hell of an accountant,
        Like all the jolly good fellows, I drink my whiskey clear,
I'm a Ramblin' Wreck from Georgia Tech and a hell of an accountant.
        Oh, if I had a daughter, sir, I'd dress her in White and Gold,
        And put her on the campus, to cheer the brave and bold.
        But if I had a son, sir, I'll tell you what he'd do.
        He would yell, "To Hell with Georgia," like his daddy used to do.
        Oh, I wish I had a barrel of rum and sugar three thousand pounds,
        A college bell to put it in and a clapper to stir it around.
        I'd drink to all good fellows who come from far and near.
I'm a ramblin', gamblin', hell of an accountant.
```

```
>>> improve_fight_song("")
```



```
I'm a Ramblin' Wreck from Georgia Tech and a hell of an engineer,
A helluva, helluva, helluva, helluva, hell of an engineer,
        Like all the jolly good fellows, I drink my whiskey clear,
I'm a Ramblin' Wreck from Georgia Tech and a hell of an engineer.
        Oh, if I had a daughter, sir, I'd dress her in White and Gold,
        And put her on the campus, to cheer the brave and bold.
        But if I had a son, sir, I'll tell you what he'd do.
        He would yell, "To Hell with Georgia," like his daddy used to do.
        Oh, I wish I had a barrel of rum and sugar three thousand pounds,
        A college bell to put it in and a clapper to stir it around.
        I'd drink to all good fellows who come from far and near.
I'm a ramblin', gamblin', hell of an engineer.
```

Function name: `my_chat`
Parameters: `sender<str>, receiver<str>, msg<str>, time<datetime.datetime>`
Return value: `None`

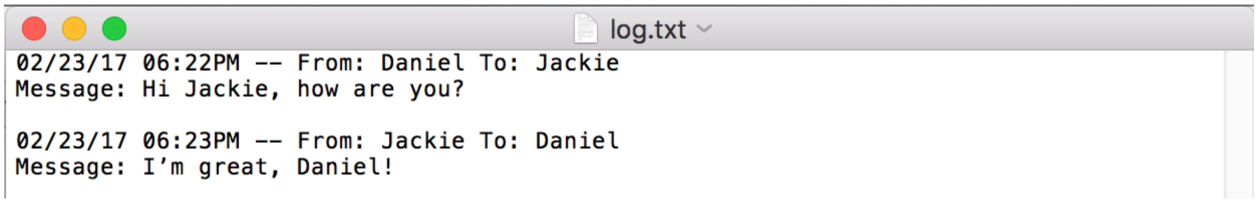
Description: Maintain an active log of a chat history by writing the information to a text file. Name your file: **log.txt**. You should write information about the message. Include the time, determined by the value passed in via the `time<datetime.datetime>` parameter. Also include the `sender<str>`, the `receiver<str>`, and finally the message, `msg<str>`. Check the test cases for exact format.

Notes:

- To access the `datetime` module you must include this import statement at the top of your script:
    - `from datetime import datetime.`
- Previous conversations should not be erased.
- The format of each line should match exactly like the examples below.
    - Obviously, your times will not match those from the examples.
- For `datetime` formatting assistance, please reference: https://docs.python.org/3/library/datetime.html#strftime-and-strptime-behavior

Test Cases:

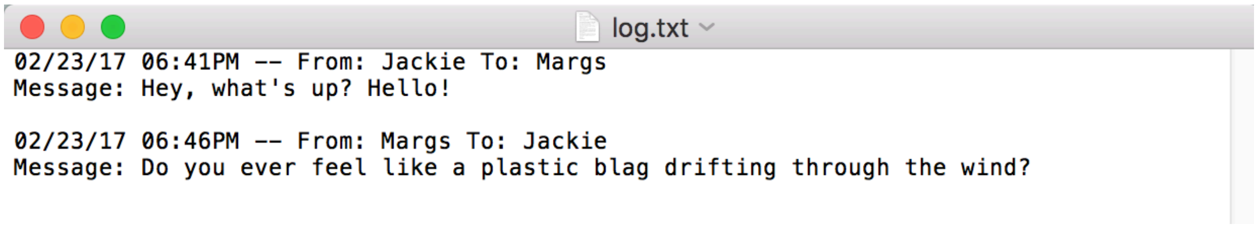
```
>>> chat("Daniel","Jackie","Hi Jackie, how are you?", datetime.now())
>>> chat("Jackie","Daniel","I'm great, Daniel!", datetime.now())
```



```
02/23/17 06:22PM -- From: Daniel To: Jackie
Message: Hi Jackie, how are you?

02/23/17 06:23PM -- From: Jackie To: Daniel
Message: I'm great, Daniel!
```

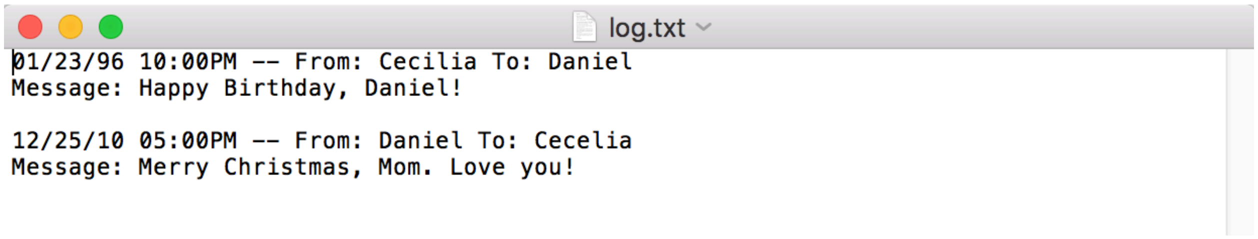
```
>>> chat("Jackie","Margs","Hey, what's up? Hello!", datetime.now())
>>> chat("Margs","Jackie","Do you ever feel like a plastic blag drifting through
the wind?", datetime.now())
```



```
02/23/17 06:41PM -- From: Jackie To: Margs
Message: Hey, what's up? Hello!

02/23/17 06:46PM -- From: Margs To: Jackie
Message: Do you ever feel like a plastic blag drifting through the wind?
```

```
>>> past_time = datetime.strptime("January 23, 1996 10:00PM",
"%B %d, %Y %I:%M%p")
>>> past_time_2 = datetime.strptime("December 25, 2010 05:00PM",
"%B %d, %Y %I:%M%p")
>>> chat("Cecilia", "Daniel", "Happy Birthday, Daniel!", past_time)
>>> chat("Daniel", "Cecelia", "Merry Christmas, Mom. Love you!", past_time_2)
```



```
01/23/96 10:00PM -- From: Cecilia To: Daniel
Message: Happy Birthday, Daniel!

12/25/10 05:00PM -- From: Daniel To: Cecelia
Message: Merry Christmas, Mom. Love you!
```

Function name: song_by_artist
Parameters: a <list> of tuples (song_title<str>, artist_name<str>, year_published<int>)
Return value: None

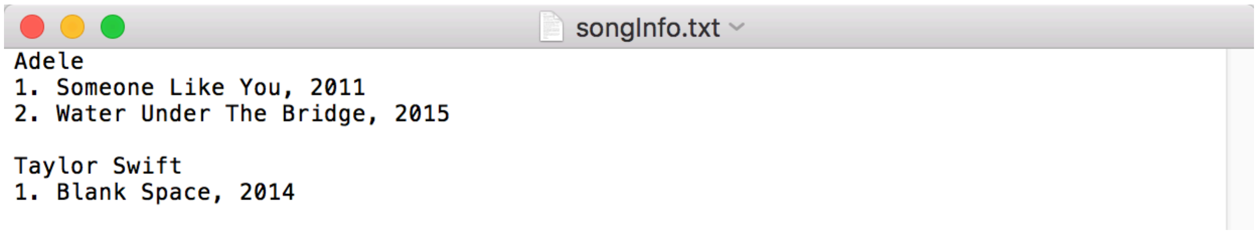
Description: Writes a file containing the information in the list sorted by artist_name<str>. The songs by an artist should be written to the file in a numbered list. The function can handle any number of tuples. The name of the file being created is **songInfo.txt**.

Notes:

- The format of the file you create should match the format below.
- You can assume the list has at least one tuple with valid components.
- Order by Artist, then by Song, then by Year.

Test Cases:

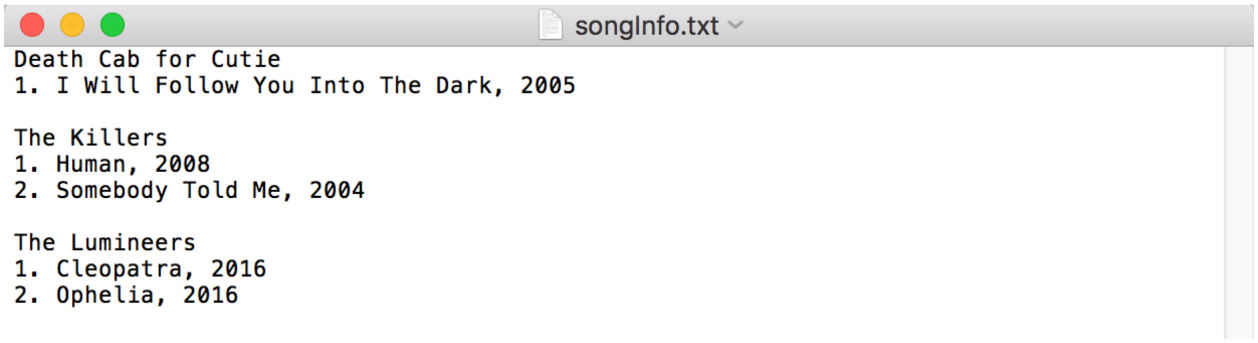
```
>>> song_by_artist([("Someone Like You", "Adele", 2011), ("Blank Space", "Taylor
Swift", 2014),("Water Under The Bridge", "Adele", 2015)])
```



```
Adele
1. Someone Like You, 2011
2. Water Under The Bridge, 2015

Taylor Swift
1. Blank Space, 2014
```

```
>>> song_by_artist([("Human", "The Killers", 2008),("Cleopatra", "The Lumineers",
2016), ("Somebody Told Me", "The Killers", 2004),("Ophelia", "The Lumineers",
2016), ("I Will Follow You Into The Dark", "Death Cab for Cutie", 2005)])
```



```
Death Cab for Cutie
1. I Will Follow You Into The Dark, 2005

The Killers
1. Human, 2008
2. Somebody Told Me, 2004

The Lumineers
1. Cleopatra, 2016
2. Ophelia, 2016
```

Part 2: CSV and Data Analysis

You have been provided with a comma-separated values file (a .csv). You can download it here: http://cs1301.com/downloads/fortune500.csv

In this section of the assignment you will be reading this file and writing functions to get statistics and data from it. In order to get some more practice with CSV in Python, I believe these resources might be helpful:

- https://pythonprogramming.net/reading-csv-files-python-3/
- https://docs.python.org/3/library/csv.html

Description of the Dataset:
The file, fortune500.csv, contains ~29.5k entries. It has the Fortune 500 ranking from the year 1955 all the way to 2009. This is a public dataset found online at (http://introcs.cs.princeton.edu/java/data/) with some minor modifications. In no way is this dataset 100% accurate of real-world data, but it has very good and interesting data we can use. It includes the following variables/columns: Year, Rank, Company, Revenue (in millions), and Profit (in millions).

Assignment:
Do not modify in any way the dataset that has been given to you as small changes might make your assignment fail test cases. For the purpose of this assignment you can assume that the strings of the company names are case sensitive. This means that the for example the company name 'Nike' is not the same as 'nike'.

Complete the following functions:

Function name: `top_companies`
Parameters: `year<int>, N<int>`
Return value: a <list> of top-N `company_names<str>`

Description: Returns a list of the top `N<int> company_names<str>` in a given `year<int>.` If N > length of list for given year, return all company_names ranked that year.
Notes:

- If a given year is not a valid year (year < 1955 or year > 2009) or if N is not a positive integer you should return an empty list.

Test Cases:

```
>>> test1 = top_companies(2007,10)
>>> print(test1)
```

```
['Wal-Mart Stores', 'Exxon Mobil', 'General Motors', 'Chevron', 'ConocoPhillips',
'General Electric', 'Ford Motor', 'Citigroup', 'Bank of America Corp.', 'American
Intl. Group']

>>> test2 = top_companies(1996, 5)
>>> print(test2)
['General Motors', 'Ford Motor', 'Exxon Mobil', 'Wal-Mart Stores', 'AT&T']

>>> test3 = top_companies(1955, 2)
>>> print(test3)
['General Motors', 'Exxon Mobil']

>>> test4 = top_companies(2015, 10)
>>> print(test4)
[]
```

Function name: `company_ranking`
Parameters: `company_name<str>, year<int>`
Return value: `tuple (Rank<int>, Revenue(millions)<float>, Profit (millions)<float>)`

Description: Returns a tuple (rank<int>, revenue(millions)<float>, profit (millions)<float>) of the company_name for the given year. Return None if the company was not ranked that year

Notes:

- Be careful converting a string to a float where that string has commas.

Test Cases:

```
>>> test1 = company_ranking('Apple Computer',2007)
>>> print(test1)
(121, 19315.0, 1989.0)

>>> test2 = company_ranking('Intl. Business Machines',1994)
>>> print(test2)
(4, 62716.0, -8101.0)

>>> test3 = company_ranking('Goldman Sachs Group',2009)
>>> print(test3)
(40, 53579.0, 2322.0)

>>> test4 = company_ranking('Starbucks',2003)
>>> print(test4)
(465, 3288.9, 215.1)

>>> test5 = company_ranking('Google',1990)
>>>print(test5)
None
```

Function name: `company_average`
Parameters: `company_name<str>`
Return value: tuple `(average<float(2decimals)>, count<int>)`

Description: Returns a tuple (average<float(2 decimals)>, count<int>) of the company_name. Where average<float> is the average ranking a given company over all the years. Where count<int> is the number of times the company has been in the ranking. Return None if the company has not been ranked any year.

Notes:

- Be careful with `ZeroDivisionError` exceptions.

Test Cases:

```
>>> test1 = company_average('Exxon Mobil')
>>> print(test1)
(2.02, 55)

>>> test2 = company_average('Apple Computer')
>>> print(test2)
(187.15, 27)

>>> test3 = company_average('Nike')
>>> print(test3)
(193.53, 15)

>>> test4 = company_average('Facebook')
>>> print(test4)
None

>>> test5 = company_average('Uber')
>>> print(test5)
None
```

Part 3: Intro to urllib

The **`urllib.request`** module defines functions and classes which help in opening URLs and getting responses from these URLs. We have written a website that will encrypt and decrypt messages for you. You will be making requests to this URLs to encrypt and decrypt messages.

Function name: `encrypt_message`
Parameters: `msg<str>`
Return value: the encrypted message `<str>`

Description: Returns an encrypted message. Get a request from the following website:

- http://cs1301.com/encrypt.php

This endpoint can take a parameter called `str` and will return an encrypted message. For example, try this in your browser, what you see in the screen will be the response.

- http://cs1301.com/encrypt.php?str=Hello

Your function should return this encrypted message.

Hints:

- Use the provided `format_url` function to get a url that is safe to use for a urllib request.
- Use the `.decode` method to convert bytes to str.
    - https://docs.python.org/3/library/stdtypes.html#bytes.decode

Function name: `decrypt_message`
Parameters: `msg<str>`
Return value: the decrypted message `<str>`

Description: Returns the decrypted message. Get a request from the following website:

- http://cs1301.com/encrypt.php

This endpoint can take a parameter called `str` and will return a decrypted message. For example, try this in your browser, what you see in the screen will be the response.

- http://cs1301.com/decrypt.php?str=rnR7GQaKESuPQt8btXfweg==

Test Cases:

```
>>> encrypted1 = encrypt_message("CS1301")
>>> print(encrypted1)
>>> print(decrypt_message(encrypted1))
D7AOGKGgWM0bo2RsvCNtQw==
CS1301

>>> encrypted2 = encrypt_message("Python")
>>> print(encrypted2)
>>> print(decrypt_message(encrypted2))
9AocgqkuD0mx2IL6CpT4fA==
Python

>>> encrypted3 = encrypt_message("Python is Great")
>>> print(encrypted3)
>>> print(decrypt_message(encrypted3))
QCrGKVbMr8GRqJ6q7a0xmw==
Python is Great
```

Part 4 (Extra Credit): Concurrent Functions

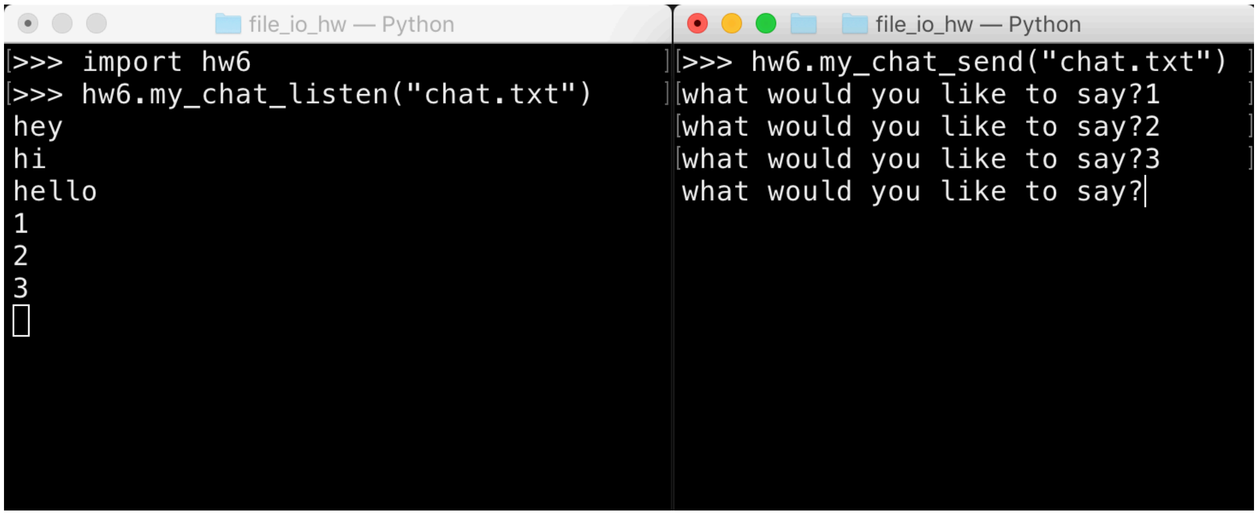
The goal here is to write two functions which will run at the same time. The first function will print the contents of a file and then continually check the file for updates. In the event that there have been lines added to the file since the last check, the function should print the new lines. To avoid excessive IO operations, you should wait for 1 second between each iteration of this infinite loop. (hint: use `time.sleep()`)

The next function, should be an infinite loop that asks the user for input. Each time the user enters a string, the function should append it to the file mentioned earlier.

Function name: `my_chat_listen`
Parameters: `filename<string>`
Return value: `None`
Description: Write a function that, on an infinite cycle, opens the file given as an argument, checks for new lines, and prints anything that has not been printed yet.

Function name: `my_chat_send`
Parameters: `filename<string>`
Return value: `None`
Description: Write a function that, on an infinite cycle, asks the user for input and writes the input to the file given as an argument.

When both of these functions are done, you can run them simultaneously in different python shells. You should be able to enter some text in `my_chat_send` and see it appear in `my_chat_listen`.



```
>>> import hw6
>>> hw6.my_chat_listen("chat.txt")
hey
hi
hello
1
2
3
```

```
>>> hw6.my_chat_send("chat.txt")
what would you like to say?1
what would you like to say?2
what would you like to say?3
what would you like to say?
```

Grading Rubric

```
- Part #1: 40 points
     - improve_fight_song:  10/40 points
     - my_chat:             10/40 points
     - song_by_artist:      20/40 points

- Part #2: 40 points
     - top_companies:   10/40 points
     - company_ranking: 15/40 points
     - company_average: 15/40 points

- Part #3: 20 points
     - encrypt_message: 10/20 points
     - decrypt_message: 10/20 points

- Part #4: [+15 points]
     - my_chat_listen & my_chat_send: 15/0 points
_____________________________________________________

- Total    115/100 points
```

Provided

The following file(s) have been provided to you. There are several, but you will only edit one of them:

1. HW6.py
   This is the file you will edit and implement. All instructions for what the methods should do are in the docstrings.

Deliverables

You must submit all of the following file(s). Please make sure the filename matches the filename(s) below. Be sure you receive the confirmation email from T-Square, and then download your uploaded files to a new folder and run them.

1. HW6.py
   If this file does not run (if it encounters an error while trying to run), you will get no credit on the assignment.