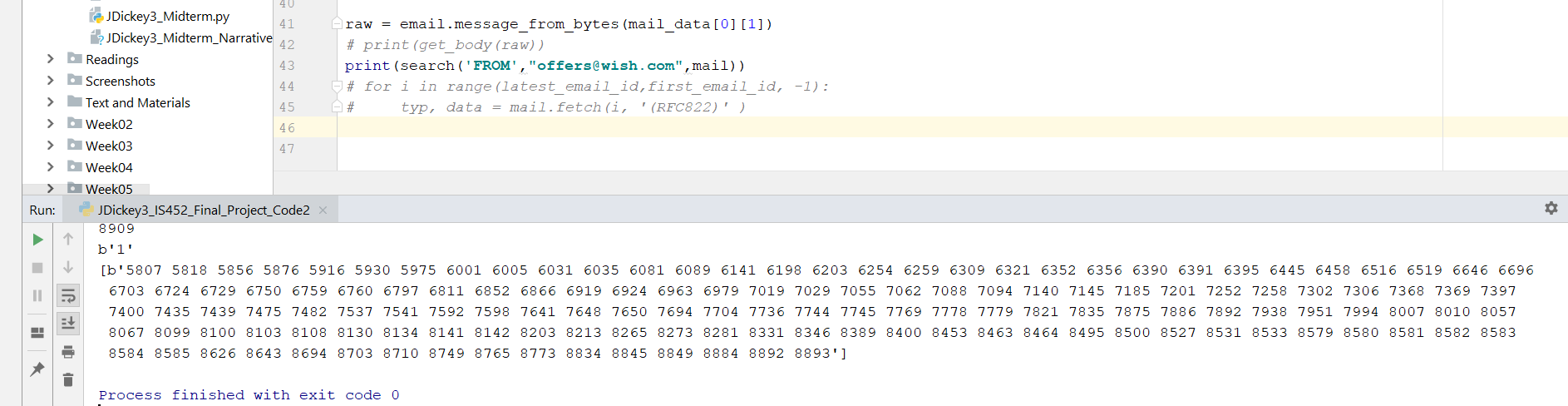
GitHub Repository: <https://github.com/jaundickey/IS452FinalProject_Fall2018>

1. **Initialization:**
   1. The tutorial for creating a GitHub found in the [IS 452 Fall 2018 Final Project Guidelines](https://courses.ischool.illinois.edu/pluginfile.php/369140/mod_assign/introattachment/0/FinalExamIS452-Fall18.pdf?forcedownload=1) was followed to create a public repository under my user account, jaundickey.
   2. Research for creating a connection to Gmail using Python programming code was performed and resulted in directions and boilerplate code obtained from the following:
      1. [Read Gmail using Python](https://pythonprogramminglanguage.com/read-gmail-using-python/), hosted by [pythonprogramminglanguage.com](https://pythonprogramminglanguage.com/).
         1. This site was the initial starting place. However, after much trial and error it appears that the boilerplate code presented is out of date (i.e. Python did not recognize “string.split()” and “rfc.822.message” as further research stated these references were discontinues with the string module now being inherent, and the RFC822 components being part of the bundle email package import.
         2. Having said that, it is very likely that the initial instructions followed in the first few steps (i.e. “enable POP support in Gmail” which provides instructions to enable IMAP in your Gmail account and then activate the setting to Access for less secure apps) allowed subsequent attempts with updated code to be successful, therefore the reference to this page remains to ensure reproducibility.
      2. [How to Read Email from Gmail using Python](https://codehandbook.org/how-to-read-email-from-gmail-using-python/), hosted by [codehandbook.org](https://codehandbook.org/how-to-read-email-from-gmail-using-python/).
         1. Leveraged boilerplate code for the utility to read email, the named function reademail(), the variable names and the structure for defining the email IDs from the list of emails
      3. [How to Read Emails using IMAP Download Attachments Puthon 3 for Beginners 2018](https://www.youtube.com/watch?v=e-OZeAHFpkw), hosted by [YouTube.com](https://www.youtube.com/).
         1. After initial unsuccessful attempts were made to generate proper server responses and email header or body content from the above boilerplate code it was determined further in-depth, step-by-step instruction was required to bring additional context for the functions and arguments used. Thus a search for audio and visual capabilities of from scratch coding was needed.
2. **Authoring Code:**
   1. Boilerplate code from 1.2.2 and 1.2.3 above were used to successfully connect to Python and obtain more information about the mail objects (see the [Connecting to Gmail Jupyter file](https://github.com/jaundickey/IS452FinalProject_Fall2018/blob/master/Connecting%20to%20Gmail.ipynb) for detailed information).
      1. Connection was a success! I was able to query a list of items (byte ids or something I think) that corresponded to a dictionary key (the FROM header) that I inserted into a custom defined search function (more boilerplate). In other words, I got a list of items resulting from my inquiry of a particular sender (in this case “offers@wish.com”).



1. **Pseudo-Code:**
   1. Continue researching connection functions and arguments to gain a very general understanding
   2. Leverage this knowledge to then create queries of the mail object data using in class concepts of dictionary (JSON evaluations) investigations, and subsequent for loops and decision trees to categorize and aggregate results according to sender and/or subject
   3. Export results, rudimentarily, into a CSV file to display aggregates or as raw data and employ SQL to summarize using grouping and/or ordering.
   4. These outputs can be used to target high volume senders to unsubscribe, pinpoint important emails from educational or financial institutions, and analyze traffic patterns between frequent contacts for networking.