\*\*\*CODE HAS BEEN MODIFIED FROM MY ORIGINAL IMPLEMENTATION TO FUNCTION IN JUPYTER HUB\*\*\*

Please see the full implementation on my public GitHub repository at: <https://github.com/tylerteichmann/RSA_Coding_Project>

#### Custom Code Feature: Decrypt Piazza

1. Introduction
   1. Problem:
      1. Decoding posts and replies one by one takes time and without a complete solution, requires multiple function calls to accomplish.
      2. Solution: all-in-one application to leverage all functions
      3. Better solution: Decode entire webpage
   2. Bigger Problem:
      1. Content is dynamically generated, traditional downloading doesn’t work
   3. Meeting the Challenge
      1. Optimization
         1. First, I implemented some optimizations within the RSA functions to create a seamless code breaking function and improve factorization time.
      2. Selenium
         1. Next, I leveraged the Selenium framework to download the Piazza webpage for scraping.
      3. Beautiful Soup
         1. Finally, I used the Beautiful Soup framework to find and replace the encrypted posts, with the decrypted messages
   4. Not a perfect implementation as there is minimal Error checking and process validation, but with the limited time and knowledge on these frameworks I feel accomplished.
2. RSA Functions
   1. Optimize FME.
   2. Optimize factorize
   3. Break\_code function
3. Selenium
   1. Downloading Webpage
      1. Tried using the python built in request module send a get request.
      2. Redirected to login screen, could not access posts
   2. Setting appropriate cookies
      1. Used the postman application to test different request headers to achieve desired result
      2. Examined request header of successful request during normal session to find required cookies.
      3. Sent cookies with request module and received login session
   3. Rendering Dynamic Content
      1. Page was ‘loading piazza’
      2. Required session to wait to download html
      3. Selenium framework
      4. Successful download of webpage (completely static)
      5. Send cookies in file, headless chrome session
   4. No error checking/testing for correct page, all visual checks atm.
4. Beautiful Soup
   1. JS to test selectors
      1. Started by running JS code in the chrome console to identify selectors for the each thread and replies
   2. Isolating public keys
      1. Lots of divs/api generated html that isn’t easy to parse
      2. Isolated strings that contained public keys using regular expressions
      3. Extract the keys in a variety of different strings
         1. Find the first number, continue adding numbers until non number
         2. Repeat process, break when two ints are found
      4. Errors if posts weren’t properly formatted, regex is limited and can produce wrong results, no checking implemented at this time.
   3. Isolating messages
      1. Similar process to public keys
      2. Slightly easier with brackets
   4. Replace the string since cyphers aren’t all in their own tag, replaces whole text content. Fine for replies but loses content in threads.
5. Demo
   1. Download html
   2. Parse content
   3. Replace data
6. Conclusion
   1. Difficult challenges
      1. A lot of new tools to learn in the Selenium and Beautiful Soup frameworks. A lot of google searches to gain understanding
   2. Rewarding Experience
      1. Was not sure if it was possible, missed the first deadline to keep trying and was able to accomplish what I wanted to do
      2. Lots of improvements can be made to portability, error checking, verboseness.
   3. New knowledge
   4. Future Implementation
      1. Web based
      2. Prompts for log in then logs user into their account to pull data
      3. Multiple posts
      4. Robust Error handling