<https://stackoverflow.com/questions/2853889/how-does-public-key-cryptography-work> # Basic info on how RSA works

<https://stackoverflow.com/questions/30056762/rsa-encryption-and-decryption-in-python> # How the Crypto.py library works but didn’t use it. Helped formulate structure of program. We almost chose to use the Crypto.py library but chose not to because it was very easy to implement RSA using it (like under twenty lines) and wanted more of a challenge

<https://github.com/JonCooperWorks> # Cryptographer and mathematician who uses Python to teach and implement different cryptographical algorithms. Used him for inverse modulus mathematics implementation and helped to structure the program layout

<https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-045j-automata-computability-and-complexity-spring-2011/lecture-notes/MIT6_045JS11_rsa.pdf> # MIT course lecture on RSA algorithm and proofs on RSA equations with an emphasis on phi function applications

<https://www.youtube.com/watch?v=QSlWzKNbKrU> # Helpful lecture to understand how RSA truly works and why it takes so long to process keys

<https://www.pygame.org/docs/> # Info on all Pygame functions

<https://www.whitman.edu/mathematics/higher_math_online/section03.08.html> # Whitman University reference index on the golden ratio (phi function)

<http://mast.queensu.ca/~ggsmith/Papers/m2schemes_book.pdf> # Queens University research paper on geometric number theory. Did not really use this much to design the program, but was an interesting read on how cryptographical algorithms can be applied using programming languages.