The RSA program is written in Python. Given a message, it can encrypt and decrypt it using an auto-generated public and private key.

For my modular exponent function, I simply used Python's built in pow(a,b,p) function.

To make sure the random numbers generated were prime, I implemented a Fermat Primality Test algorithm.

I believe the entire program works correctly. To make sure the program was correct, I implemented an if statement that would make sure the mod of the two prime numbers was 1. If it wasn't, the program would run indefinitely. This is where I ran into an issue, the random number generator being called inside the generate prime. I was overthinking the random number generator, trying to get it to be in the right range. I tried using getrandbit until I realized I could just do randint(10**100, 10**150).

To run the program, simply execute the main.py file with a message.txt file in the same directory. The text file should contain only numbers; no spaces or enters should be in the file. The program will generate its own files for the private key, public key, and ciphertext.

Once started, it will first generate the public and private keys. After that, you can type e to encrypt the message, d to decrypt, or q to quit.

*Note: All text files, except for the message.txt file, get regenerated each run.