Project 3 on RSA:

In your favorite language (preferable in Python) create the following functions:

1. *MRT* ➔Use Miller-Rabin Primality Test to choose prime number with s=512 bits and check the primality test.
2. *EA* ➔Use Euclidean Algorithm to evaluate gcd
3. *EEA* ➔Use Extended Euclidean Algorithm to find modular inverse of the value
4. *powmod_sm*➔ Square and multiply algorithm to evaluate exponentiation.

Now write the code for

I. RSA Key Generation (use above functions 1., 2., 3.) should be
   a. Choose two primes p and q of s bits using *MRT* where p is not equal to q.
   b. Calculate $n = p * q$, and $\phi(n) = (p-1) * (q-1)$
   c. Chose randomly e from the set of $\{1,\ldots, \phi(n)-1\}$ and check using EA if $gcd(e, \phi(n)) = 1$ if not chose again until it full fills the condition.
   d. Calculate $d = e^{-1} \ mod \ \phi(n)$ using EEA. Note that d should be at least $0.3 * s$ bits
   e. Output $k_{Pub} = (n, e)$ and $k_{Pr} = (d)$

II. RSA Encryption with input $k_{Pub} = (n, e)$ and random plaintext x and output should be ciphertext y, evaluate exponentiation using the function *powmod_sm*

III. RSA Decryption with input $k_{Pr} = (d)$ and ciphertext y and output should be plaintext x, evaluate exponentiation using the function *powmod_sm*. Please make sure to check that you get the same plaintext value before the encryption.