ASSIGNMENT 2

Aim: Write a program in C++ or Java to implement RSA algorithm for key genention and cipher verification.

Theory:

Asymmetric / Public Key Algorithm:

Public Key algorithms were evolved to solve the problem of key distribution in symmetric algorithms. This is achieved by using one key for encryption and a different but related key for decryption. These algoritms are designed such that it is computationally infeasible to ~~det~~ determine the decryption key given only knowledge of the cryptographic algorithm and the encryption key. Also in some algorithms, such as RSA, either of the two related keys can be used for encryption, with the other used for decryption.

The essential steps in public key algorithm are as follows:

1. Each user generates a pair of keys to be used for the encryption and decryption of messages.
2. Each user places one of the two keys in a public register or the other accessible file. This is the public key. The companion key is

kept private. Each user maintains a collection of public keys obtained from other parties participating in communication.

3. If A wishes to send a confidential message to B, A encrypts the message using B's public key.
4. When B receives the message, B decrypts it using B's private key. No other recipient can decrypt the message because only B knows B's private key.

RSA Algorithm:

RSA (Rivest, Shamir, Adleman who first publicly described it), an algorithm for public-key cryptography involves three steps key generation, encryption and decryption.
RSA is a block cipher with each block having a binary value less than some number n. That is the block size must be less than or equal log(n). Encryption and decryption are of the following form, for some plaintext block M and ciphertext block C:

$$C = M^e \bmod n$$
$$M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$$

Both sender and receiver must know n, the sender knows the value of e, and only the receiver knows the value of d. Thus, this is a public-key encryption algorithm with a public key of PU = {e, n} and a private key of PR = {d, n}. For this algorithm to be satisfactory for public key encryption, the following requirements must meet:

1. It is possible to find values of e, d, n such that $M^{ed} = M \bmod n$ for all $M < n$
2. It is relatively easy to calculate M^e and C^d for all values of $M < n$.
3. It is infeasible to determine d given e and n.

Algorithm:

I. Key generation:

The keys (public key and private key) for the RSA algorithm are generated as:

1. Choose two distinct prime numbers p and q.
2. Compute $n = p * q$.
3. Compute $\phi(n) = (p-1)(q-1)$
4. Choose an integer e such that $1 < e < \phi(n)$ and $\gcd(e, \phi(n)) = 1$ i.e e and $\phi(n)$ are co-prime.

~~5. Determine d = e-1 mod(~~

5. Determine $d = (e^{-1}) \bmod \phi(n)$

Public ~~key~~ key : $PU = \{e, n\}$
Private key : $PR = \{d, n\}$

II. Encryption:

Alice transmits her public key (e, n) to Bob and keeps the private key ~~pri~~ secret. Bob then ~~th~~ wishes to send message M to Alice. He computes the cipher text C 4) corresponding to

$$C = M^e \bmod n$$

This can be done quickly using the method of exponentiation by squaring.

III. Decryption:

Alice can recover M from C by using her private key exponent d via computing 5)

$$M = C^d \bmod n$$

Conclusion: RSA algorithm was successfully implemented ~~for~~ text input in C++ and Java.