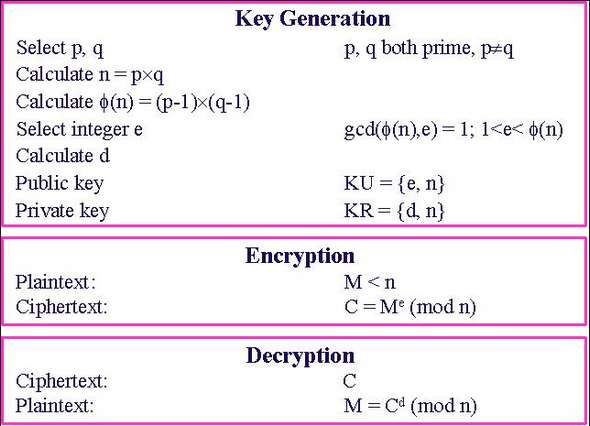
**Information Privacy and Security Topic: RSA Implementation**

**RSA** is one of the first practicable [public-key cryptosystems](http://en.wikipedia.org/wiki/Public-key_cryptography) and is widely used for secure data transmission. In such a [cryptosystem](http://en.wikipedia.org/wiki/Cryptosystem), the [encryption key](http://en.wikipedia.org/wiki/Encryption_key) is public and differs from the [decryption key](http://en.wikipedia.org/wiki/Decryption_key) which is kept secret. In RSA, this asymmetry is based on the practical difficulty of [factoring](http://en.wikipedia.org/wiki/Factorization) the product of two large [prime numbers](http://en.wikipedia.org/wiki/Prime_number), the [factoring problem](http://en.wikipedia.org/wiki/Factoring_problem). RSA stands for [Ron Rivest](http://en.wikipedia.org/wiki/Ron_Rivest), [Adi Shamir](http://en.wikipedia.org/wiki/Adi_Shamir) and [Leonard Adleman](http://en.wikipedia.org/wiki/Leonard_Adleman), who first publicly described the algorithm in 1977. [Clifford Cocks](http://en.wikipedia.org/wiki/Clifford_Cocks), an English mathematician, had developed an equivalent system in 1973, but it wasn't [declassified](http://en.wikipedia.org/wiki/Classified_information) until 1997.

A user of RSA creates and then publishes a public key based on the two large [prime numbers](http://en.wikipedia.org/wiki/Prime_number), along with an auxiliary value. The prime numbers must be kept secret. Anyone can use the public key to encrypt a message, but with currently published methods, if the public key is large enough, only someone with knowledge of the prime numbers can feasibly decode the message. Breaking RSA [encryption](http://en.wikipedia.org/wiki/Encryption) is known as the [RSA problem](http://en.wikipedia.org/wiki/RSA_problem). It is an open question whether it is as hard as the factoring problem.

–Source: *Wikipedia, (2014). RSA (cryptosystem). [online] Available at: http://en.wikipedia.org/wiki/RSA\_%28cryptosystem%29 [Accessed 10 Nov. 2014].*

**RSA Algorithm**

****

**Description of program:**

The program allows a user to select the q and q from two drop down list and then generate the encryption (e) key and decryption (d) key to encrypt a plain text file. Allows the file to be encrypted the file using either the e or the d key, the decryption is vice-versa depend on the encryption. Only input is the plain text. After the plain text is encrypted, then the cipher texts are created, the decrypted files are created after the cipher text is decrypted.

**Program Objectives:**

1. This program is for demonstrative purposes only, to make RSA more secure large prime numbers must be used and also padding is required for it to be unrecognisable.
2. To generate a cipher text where the message to encrypt is larger than p times q. After encrypting (with e), decrypt to see if the decrypted cipher was the same as the original plaintext.
3. Verify that if I use the “e” encryption and “d” decryption vice-versa would produce identical cipher text and original plain text.
4. Verify that if “q” and “p” are close in range then the original plain text could be recovered from the ciphered text. It was never mention why q and p can’t be the same number or close in range. This something I would have to look into.
5. These are the question I am left to answer at a later time:

* Why the algorithm does not work for close p and q?
* Why does m has to be smaller than “n”.

**Input:** Plain Text File to be encrypted. File name:

* plain text.txt to be encrypted.

**Outputs:** Two Cipher Texts and Two Decrypted Plain Text.

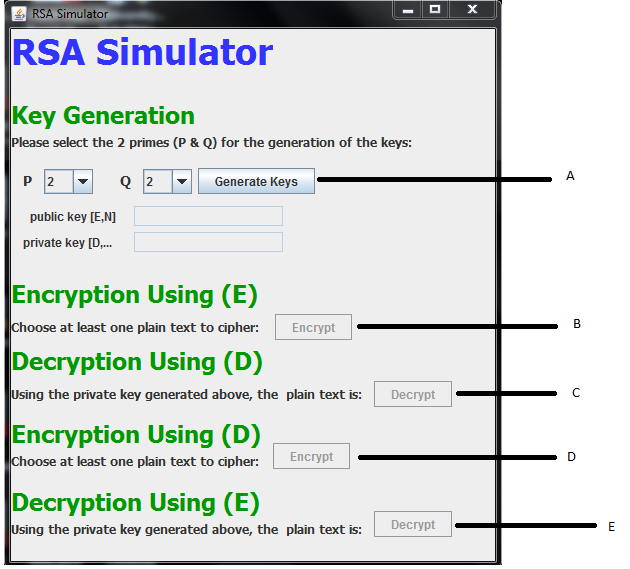
* cipherText.txt stores the encrypted plain text using the “e” as the encryption key.
* cipherText2.txt stores the encrypted plain text using the “d” as the encryption key.
* decryptText.txt stores the decrypted cipher text using the “d” as the decryption key.
* decryptText2.txt stores the decrypted cipher text using the “e” as the decryption key.

**Case 1:** Chose q and p to be relative small, so that “n” is also small compared to the message. Observe the plain text, and the decrypted text.

**Case 2:** Chose q and p to be the value same, could large or small. Observe the plain text and decrypted text.

**Case 3:** Chose q and p to be large and has a large range. Observe the plain text and decrypted text.

**Program Functions:**



**A:** is used to generate the encryption and decryption keys.

**B:** Encrypt the plain text using the “e”.

**C:** Decrypt the cipher text using the generated “d”.

**D:** Encrypt the plain text using the generated “d”.

**E:** Decrypt the plain text using the generated “e”.