**Cryptography Project Report**

Vorster Naude

Brett

Benedict Assan 24514667

North West University

Contents

**No table of contents entries found.**

### Vigenere Cipher

The Vigenere Cipher is an alphabetic encryption method using a series of interwoven Caesar’s based on the letters of a set keyword, making it a form of polyalphabetic substitution. The idea behind the cipher is to disguise plaintext letter frequencies to interfere with a straightforward application of frequency analysis.

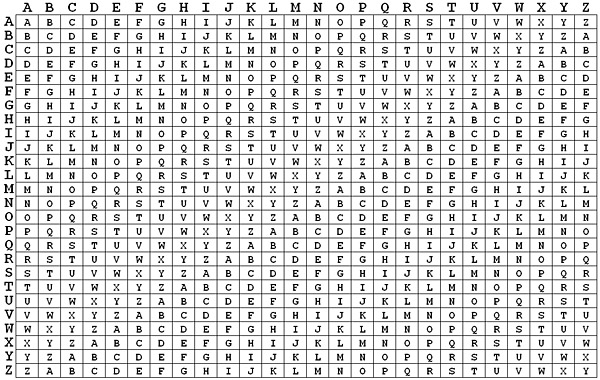
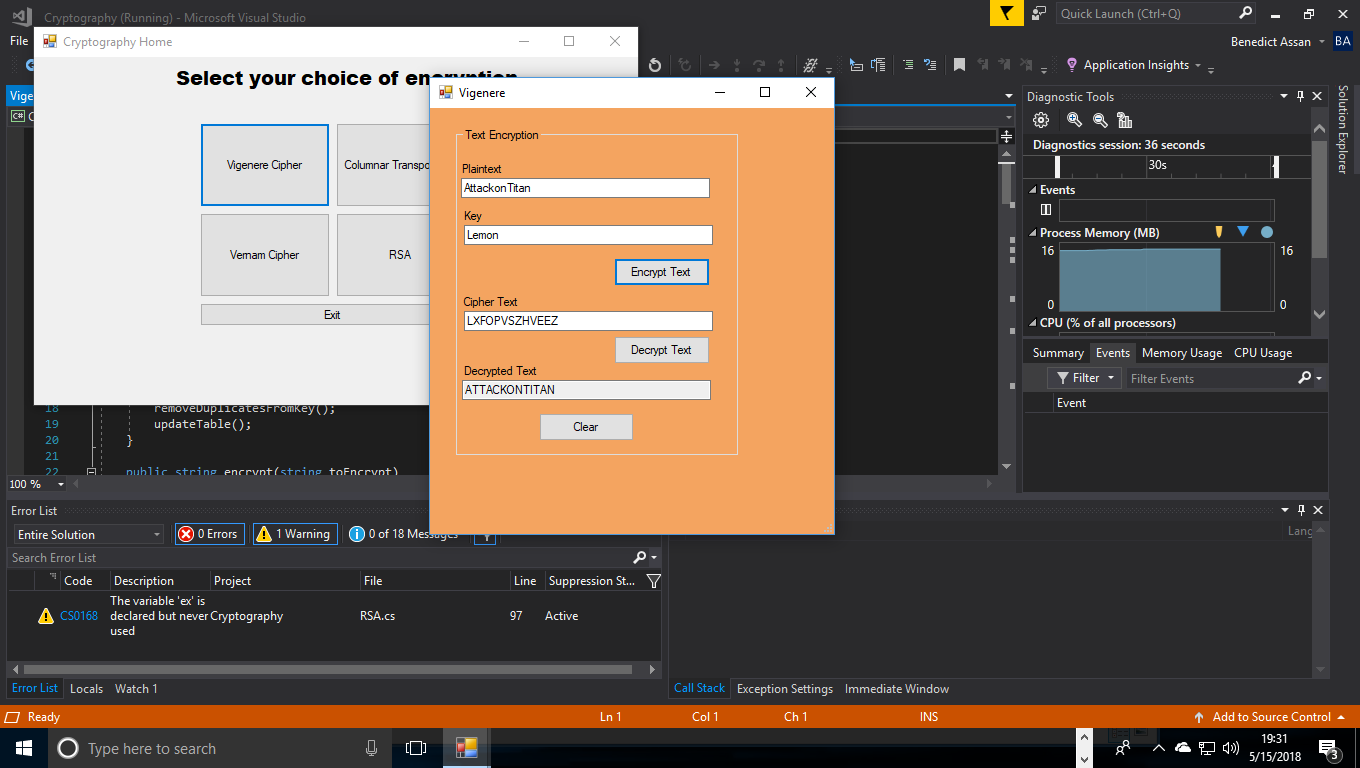


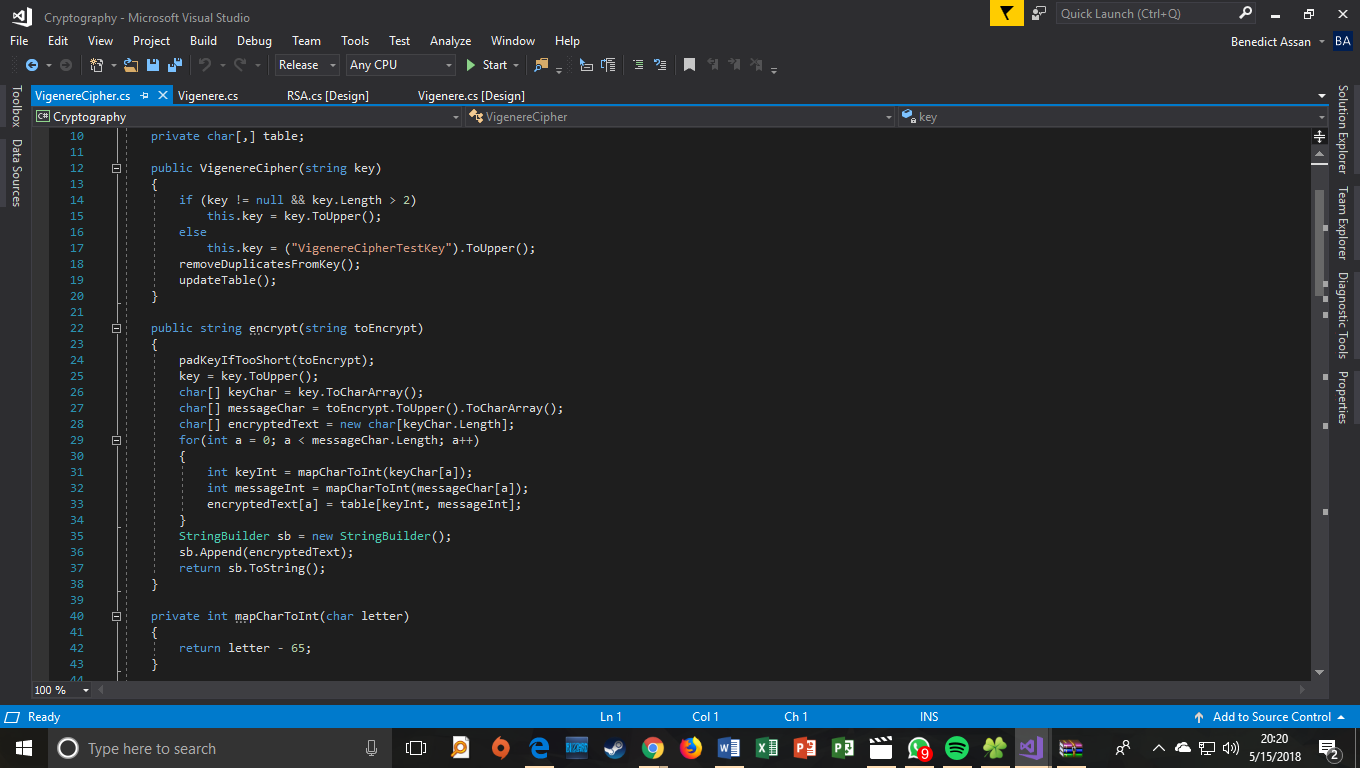
Image available at: https://pages.mtu.edu/~shene/NSF-4/Tutorial/VIG/Vig-Base.html

When looking at a Vigenere table, each row will start with a key letter and the remainder of the row holds the letters from A to Z. Despite there being 26 keys being shown, for every successive letter of the plaintext message, each successive letter of the key word enciphers each letter using the corresponding key rows.

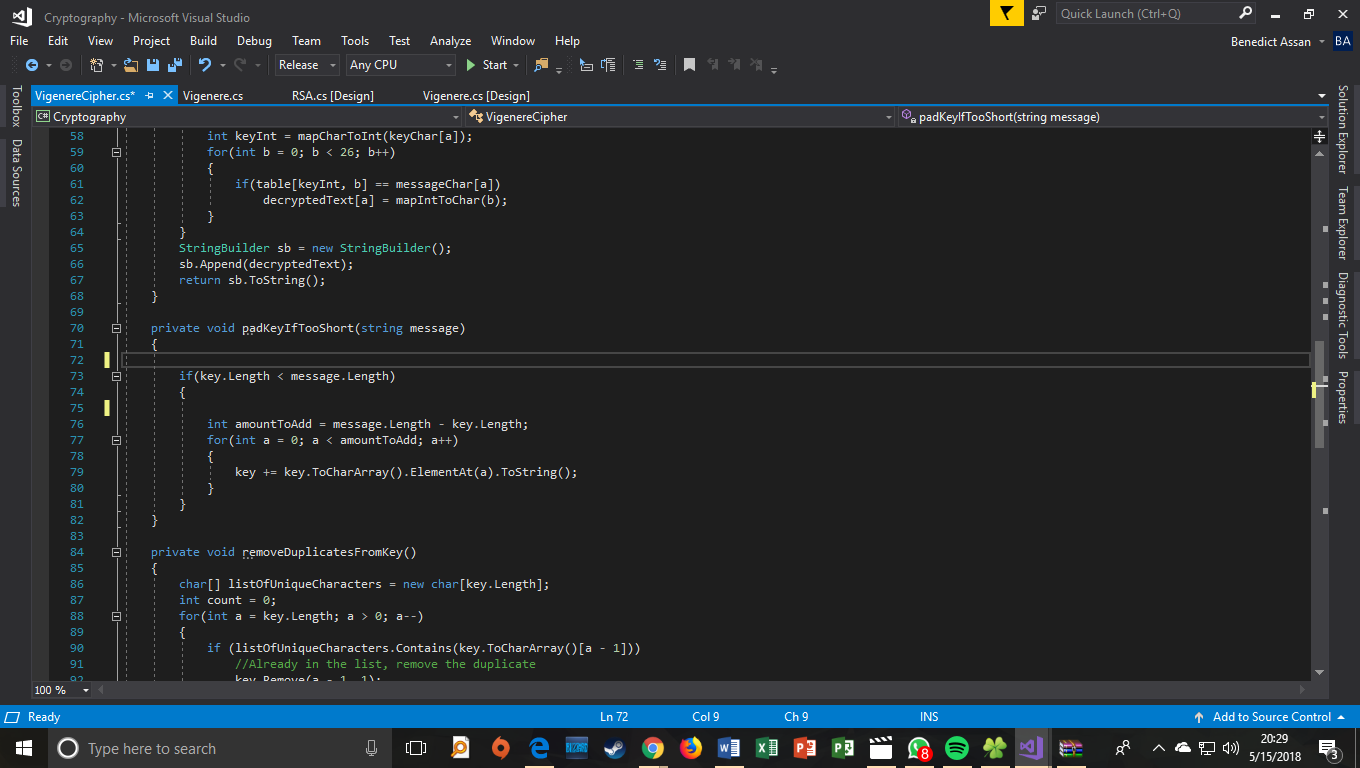


As displayed in the program above, the first letter of the plaintext ‘A’ is paired with ‘L’ from using row ‘L’ and Column ‘A’ of the Vigenere square. The second letter is encrypted the same way using row ‘E’ and column ‘T’ which contains the ciphertext ‘X’ and so on. The user just merely has to enter a plaintext message in the ‘Plaintext’ box and their chosen keyword and click on the ‘Encrypt Text’ button to create the ciphertext.

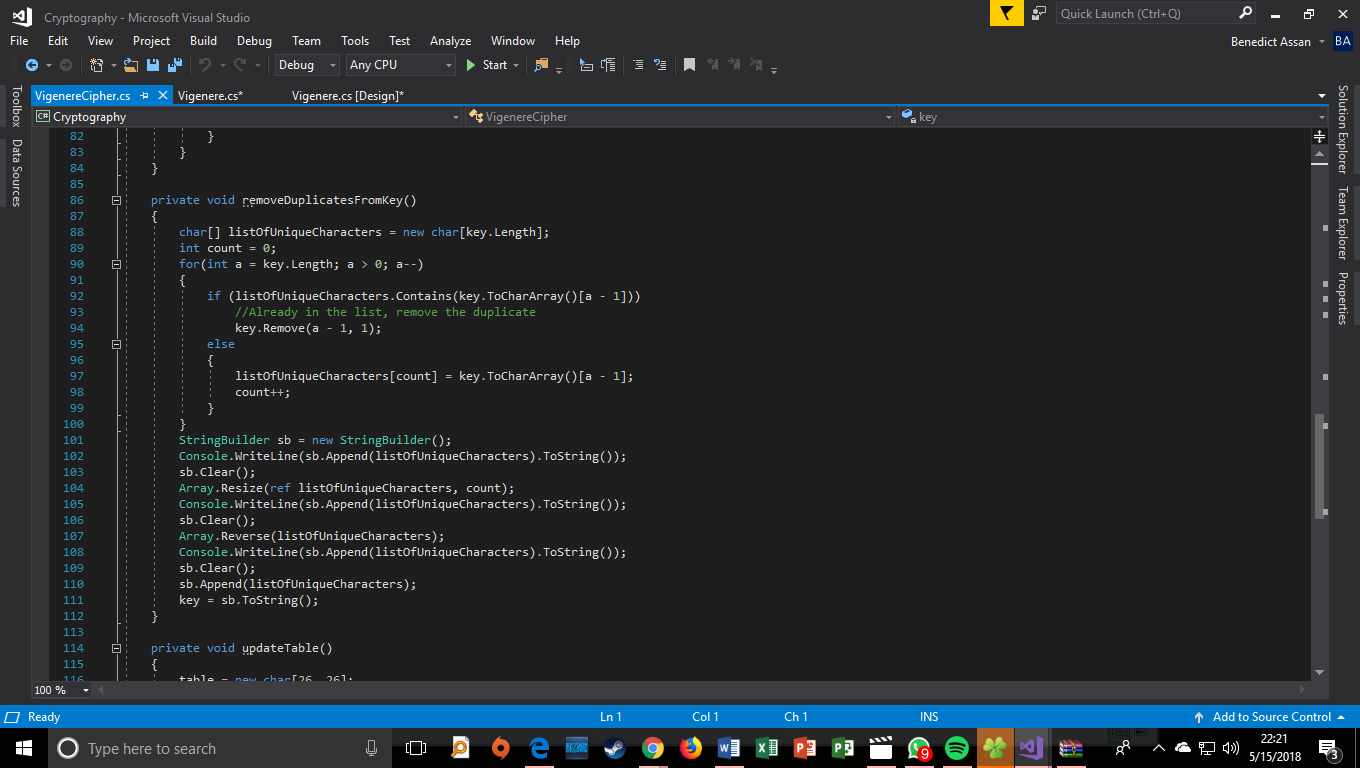
Encryption is made possible with the following method:



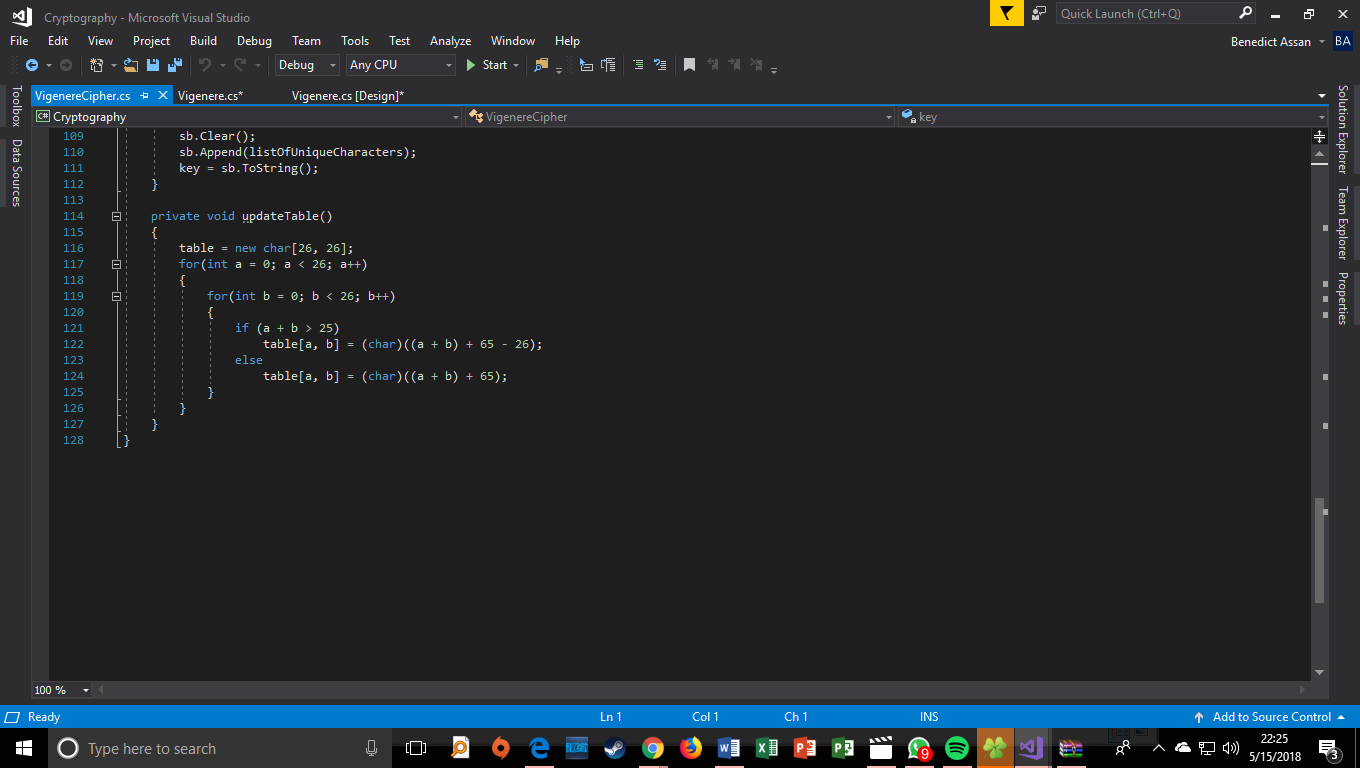
If the key is too short for the plaintext message, a padding method is used to supplement the ciphertext:



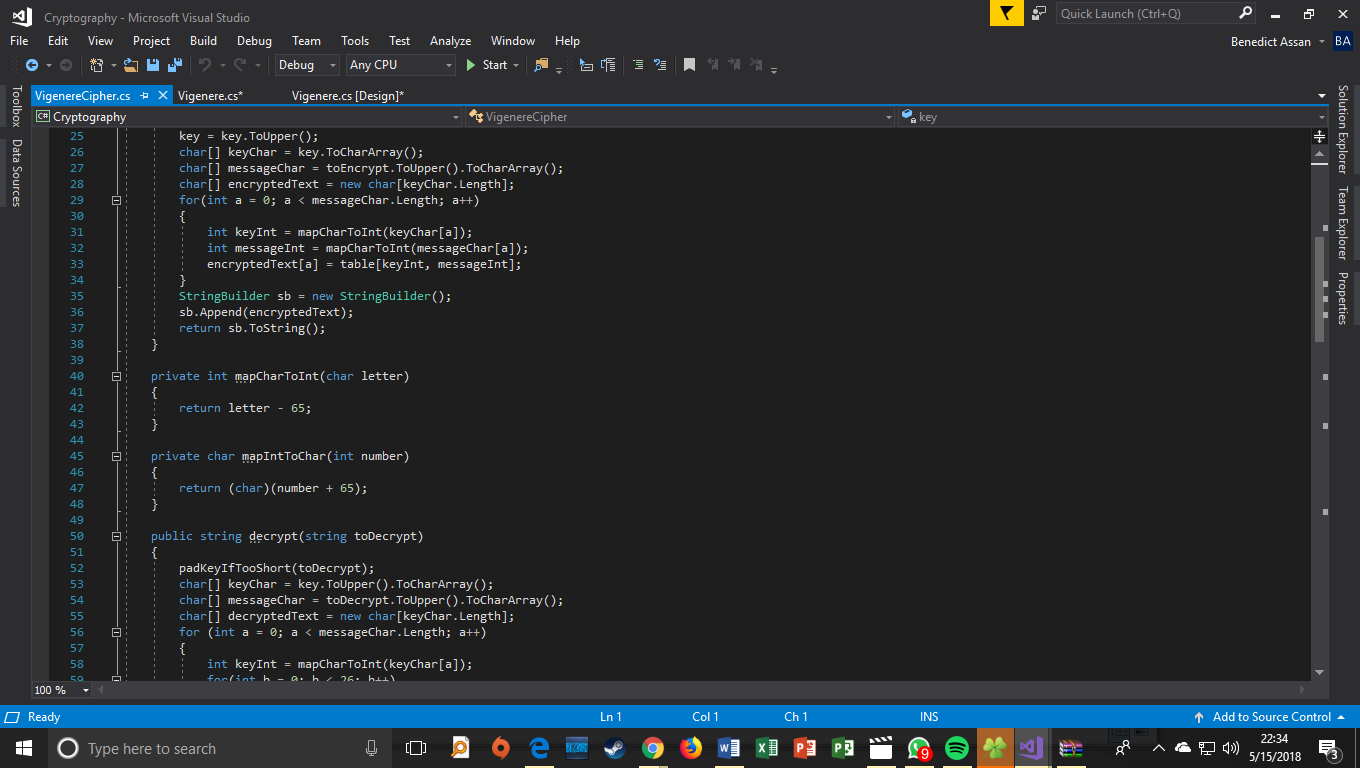
If the key uses any repeating letters, a method is used here to remove duplicate letters from the key:



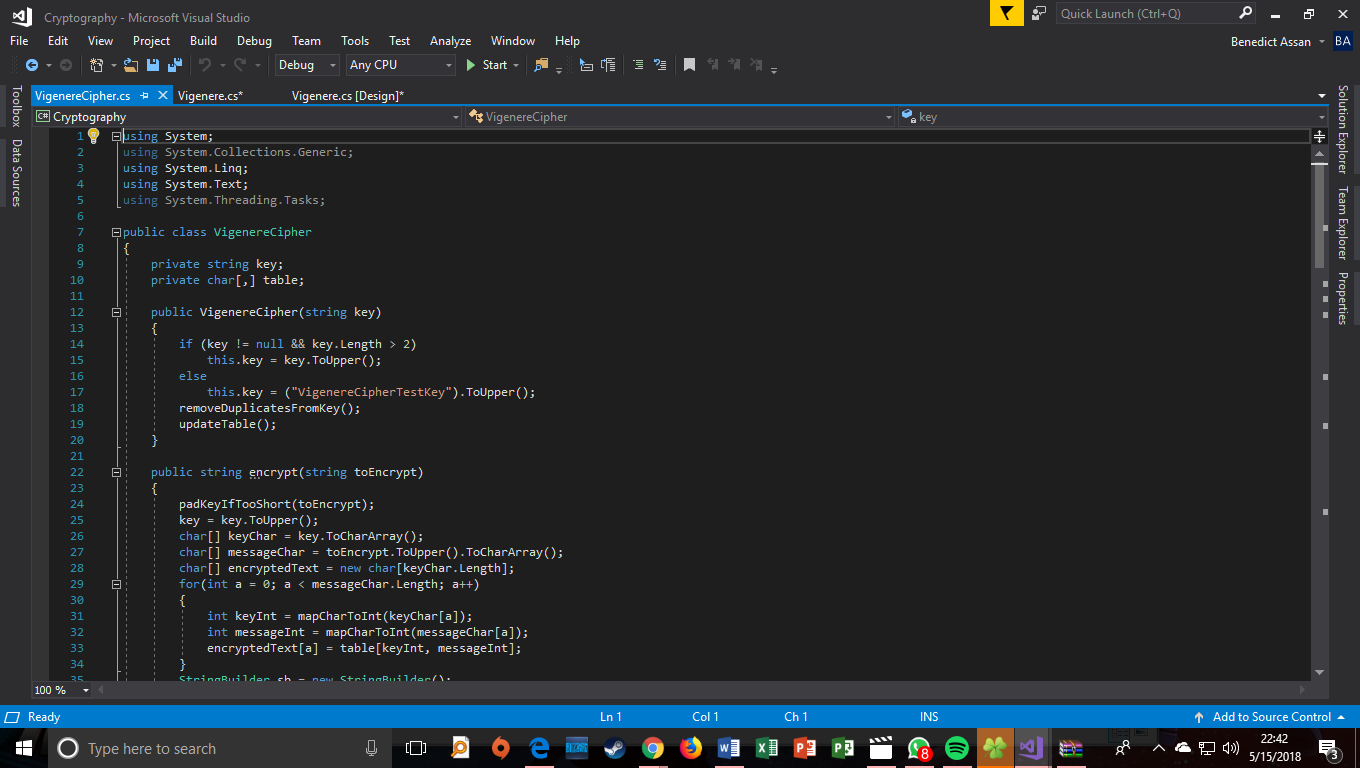
The Vigenere table method.



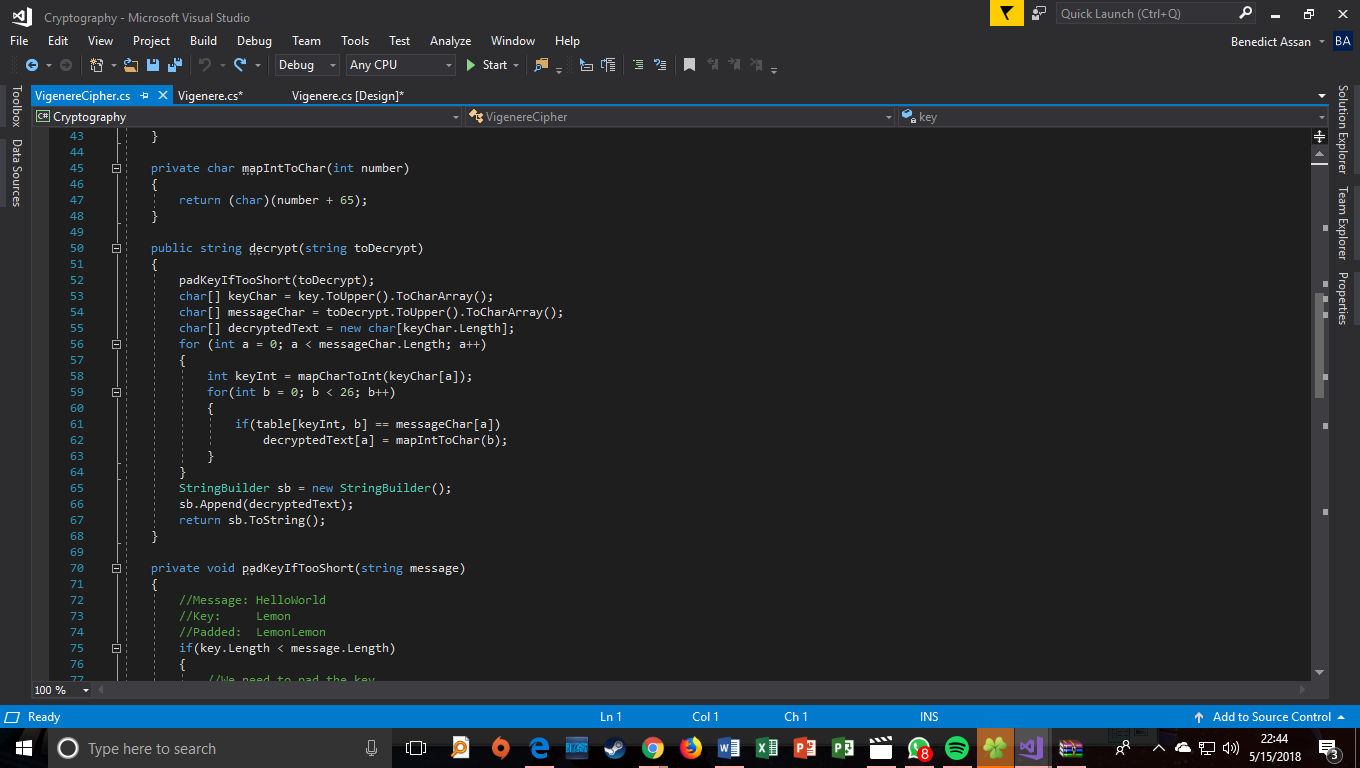
These two methods map the total integer values in the table to the number of characters present in the Vigenere table:



This method activates the cipher:

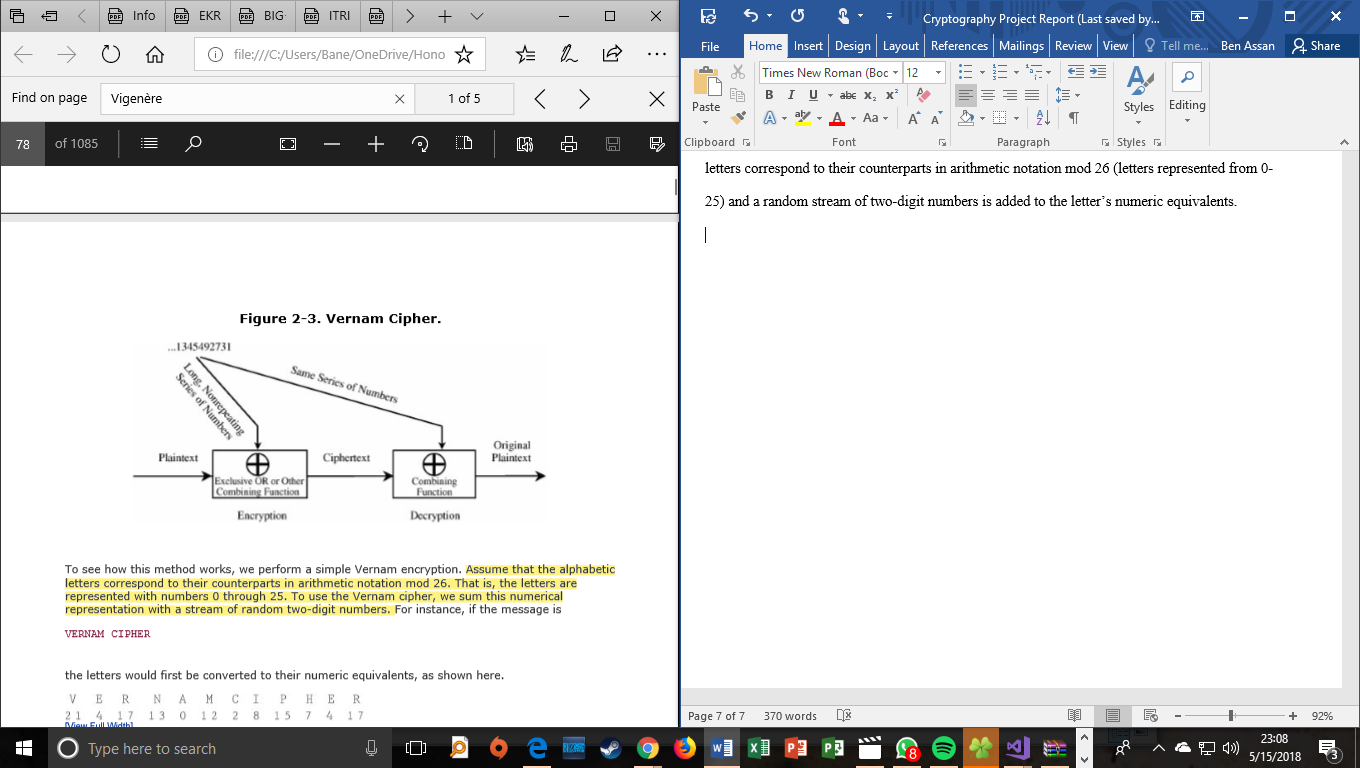


And finally, decryption is done through this method:

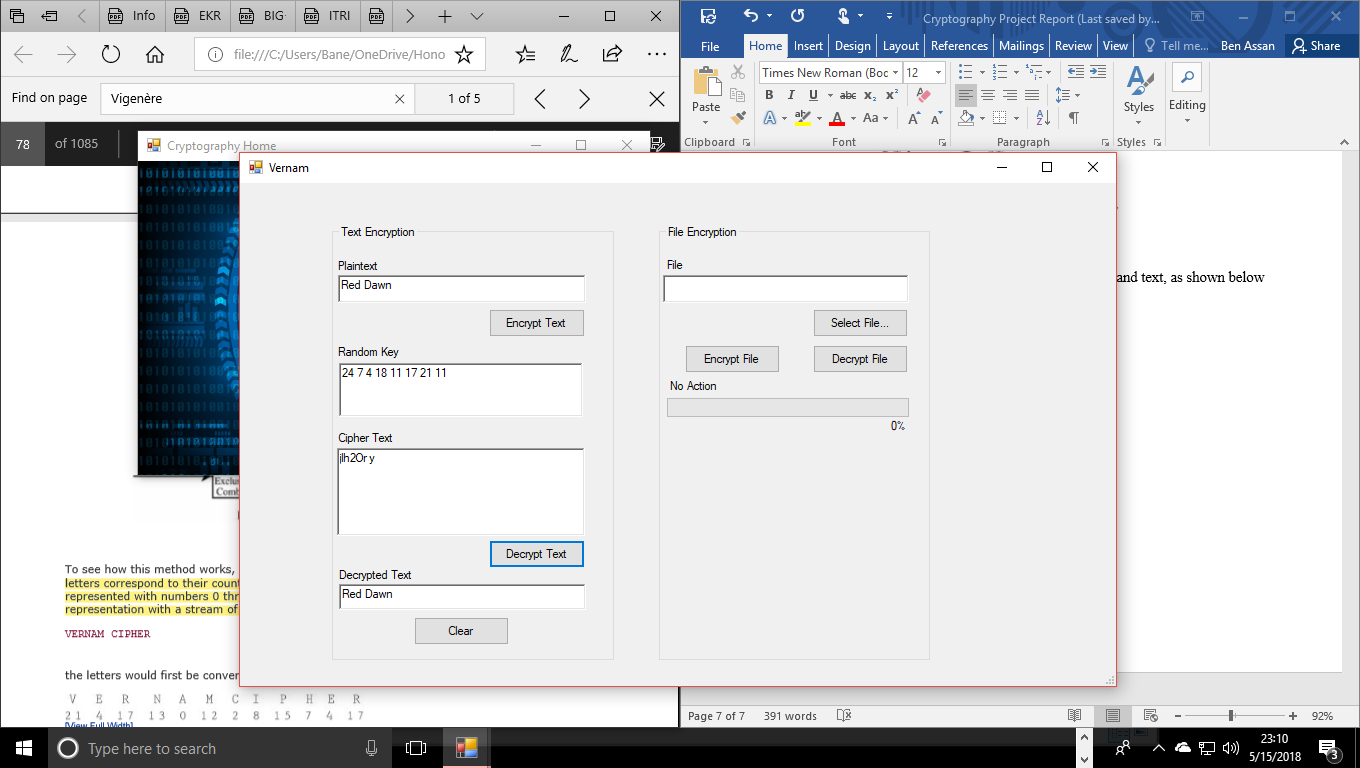


### Vernam Cipher (One-time pad)

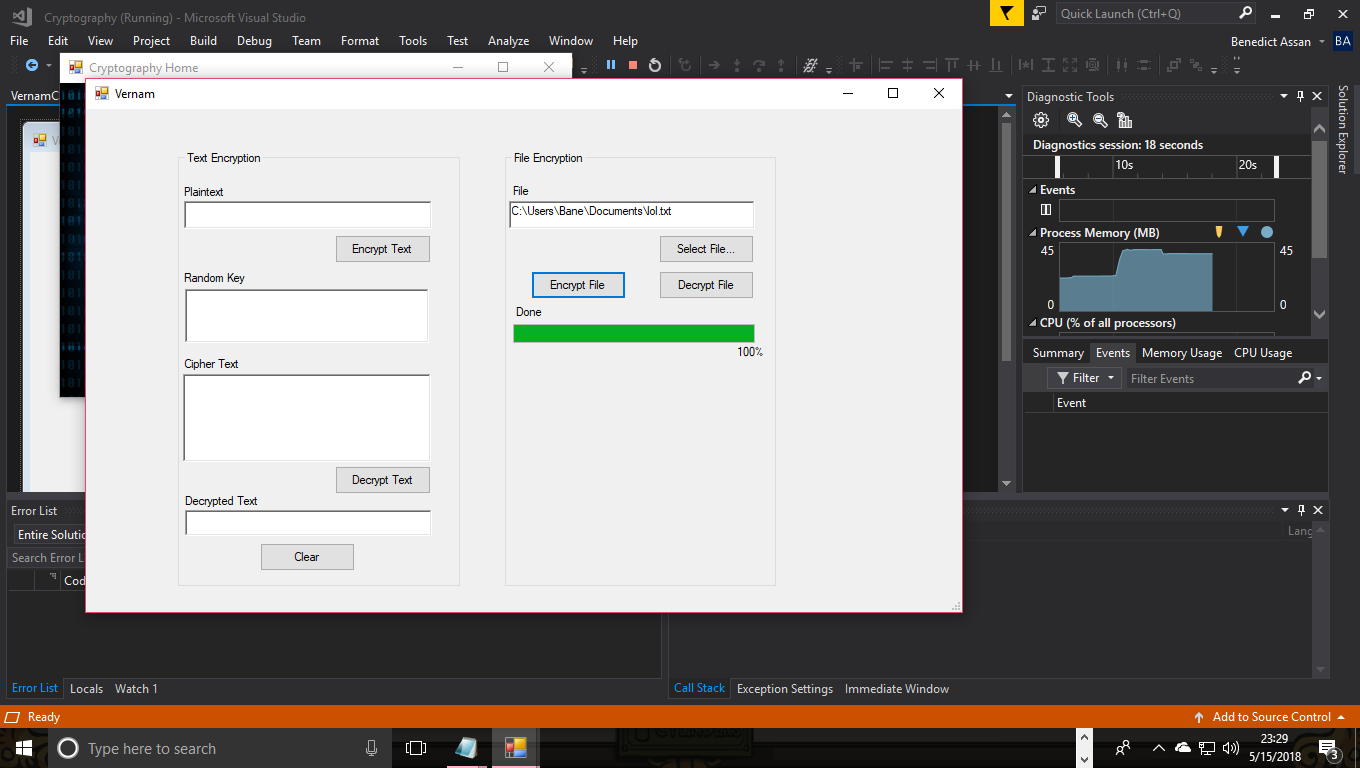
The Vernam cipher is a type of one-time pad immune to most cryptanalytic attacks and is described as the ‘perfect cipher.’ Encryption basically involves an arbitrarily long nonrepeating sequence of numbers combined with the plaintext. The cipher works by assuming the alphabetic letters correspond to their counterparts in arithmetic notation mod 26 (letters represented from 0-25) and a random stream of two-digit numbers is added to the letter’s numeric equivalents.



The program we have developed uses this cipher to encrypt both files and text, as shown below with the text encryption:

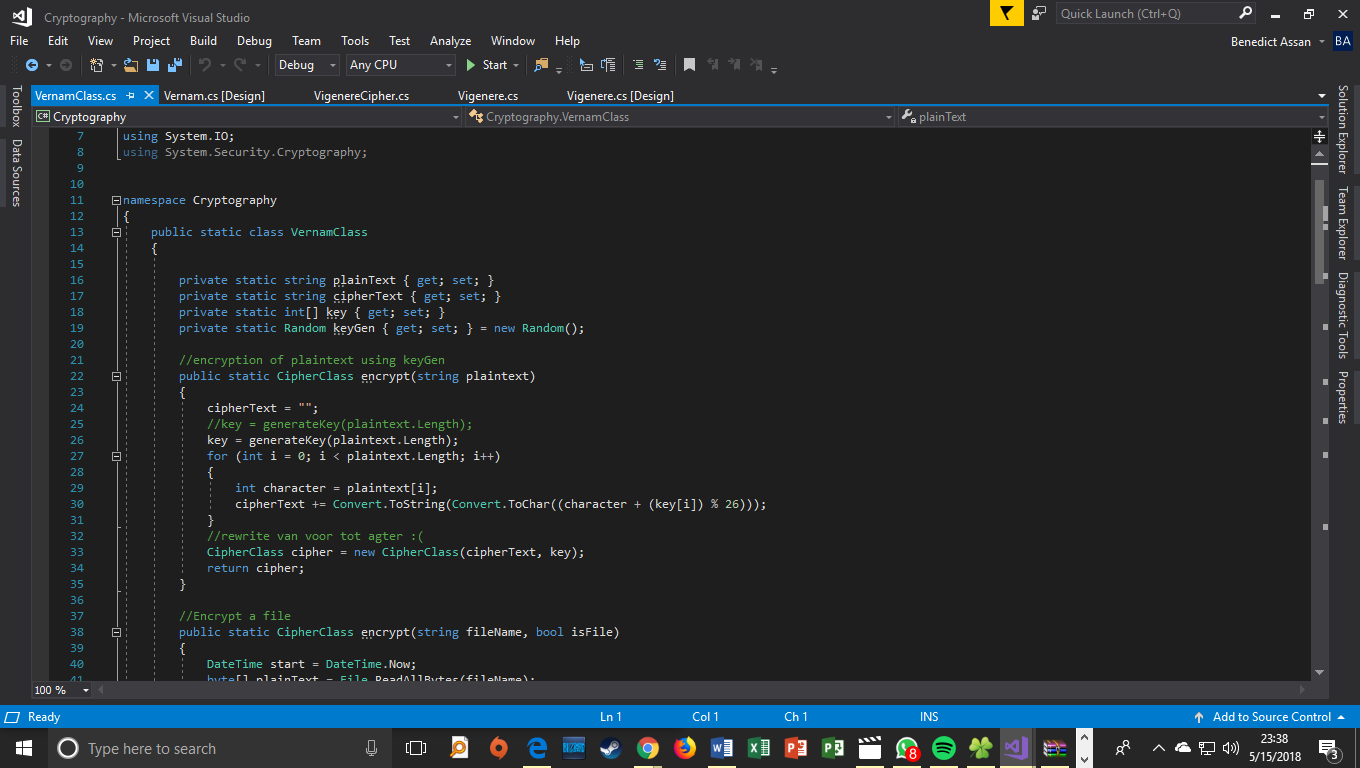


And file encryption:

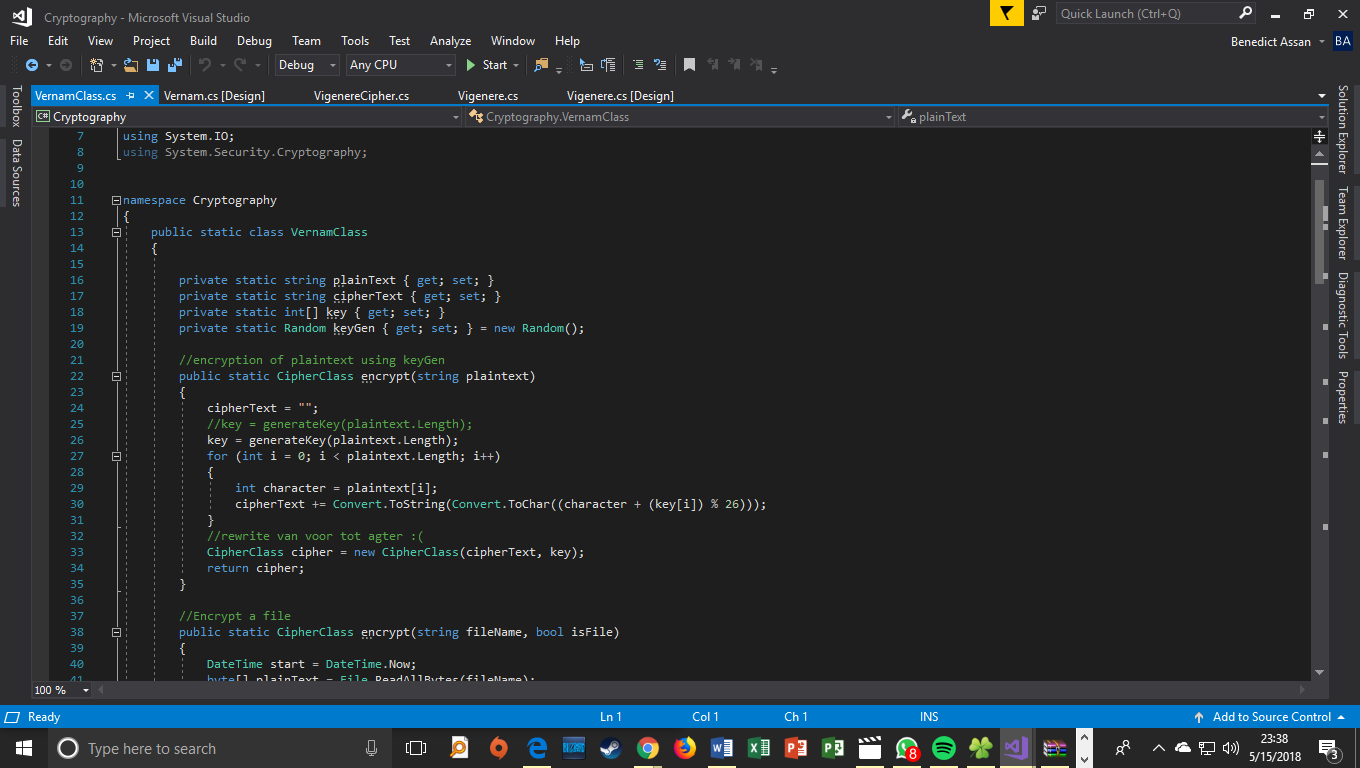


The system creates the encrypted and decrypted files on the user’s file system.

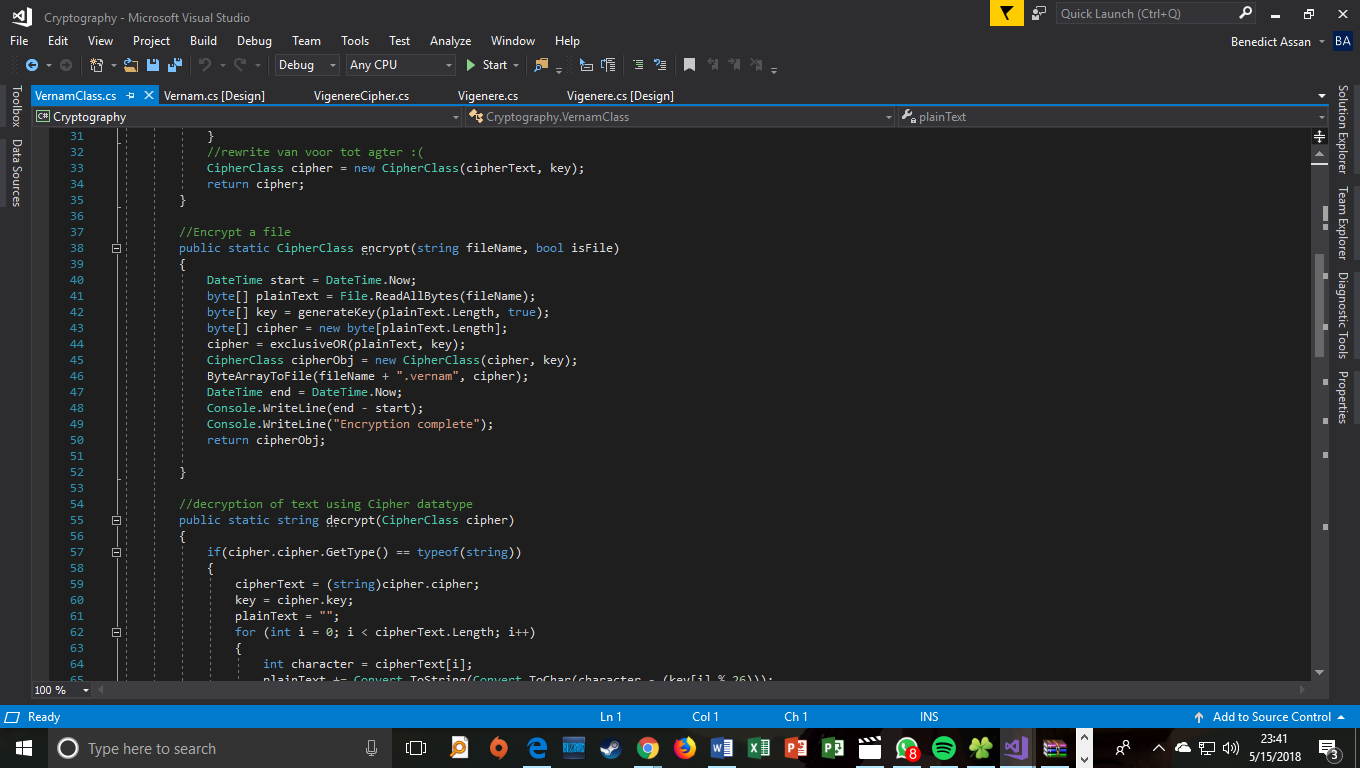
Global variables:



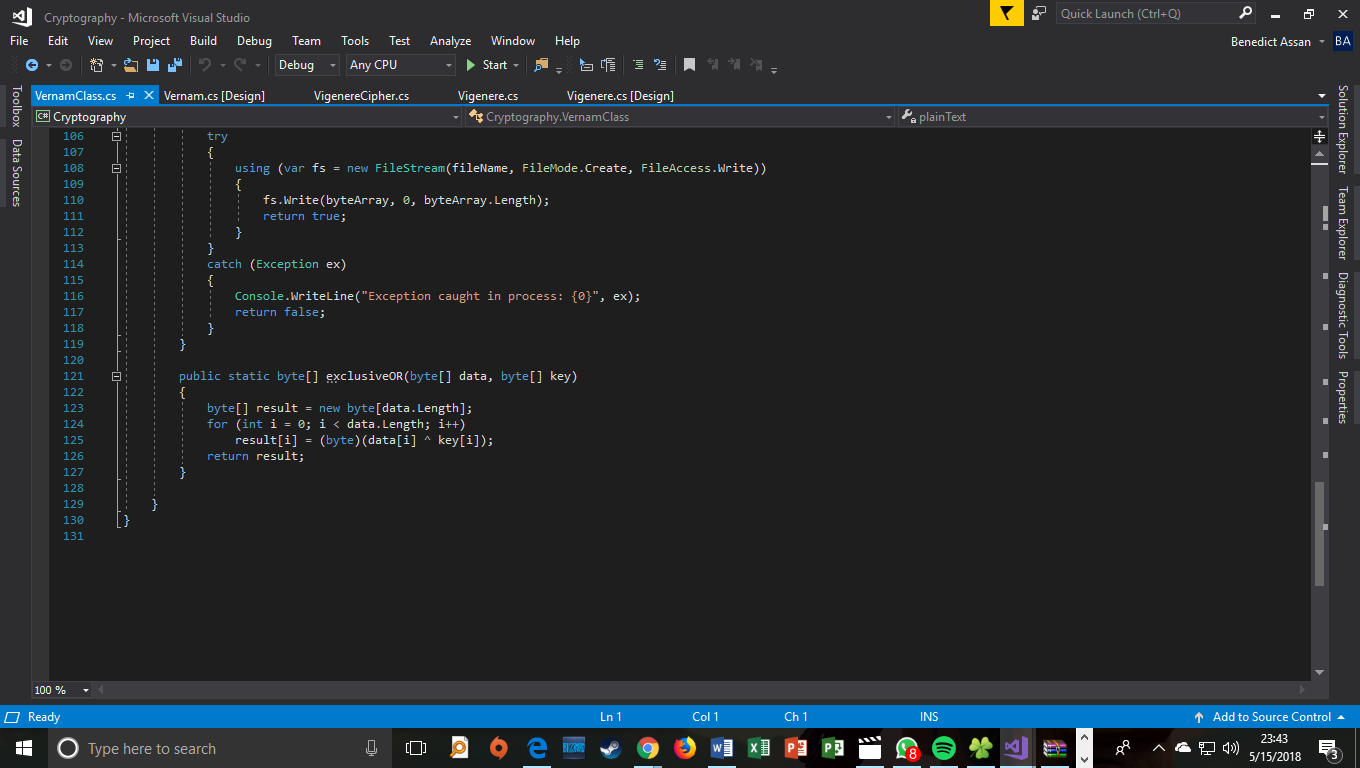
Plaintext encryption method:



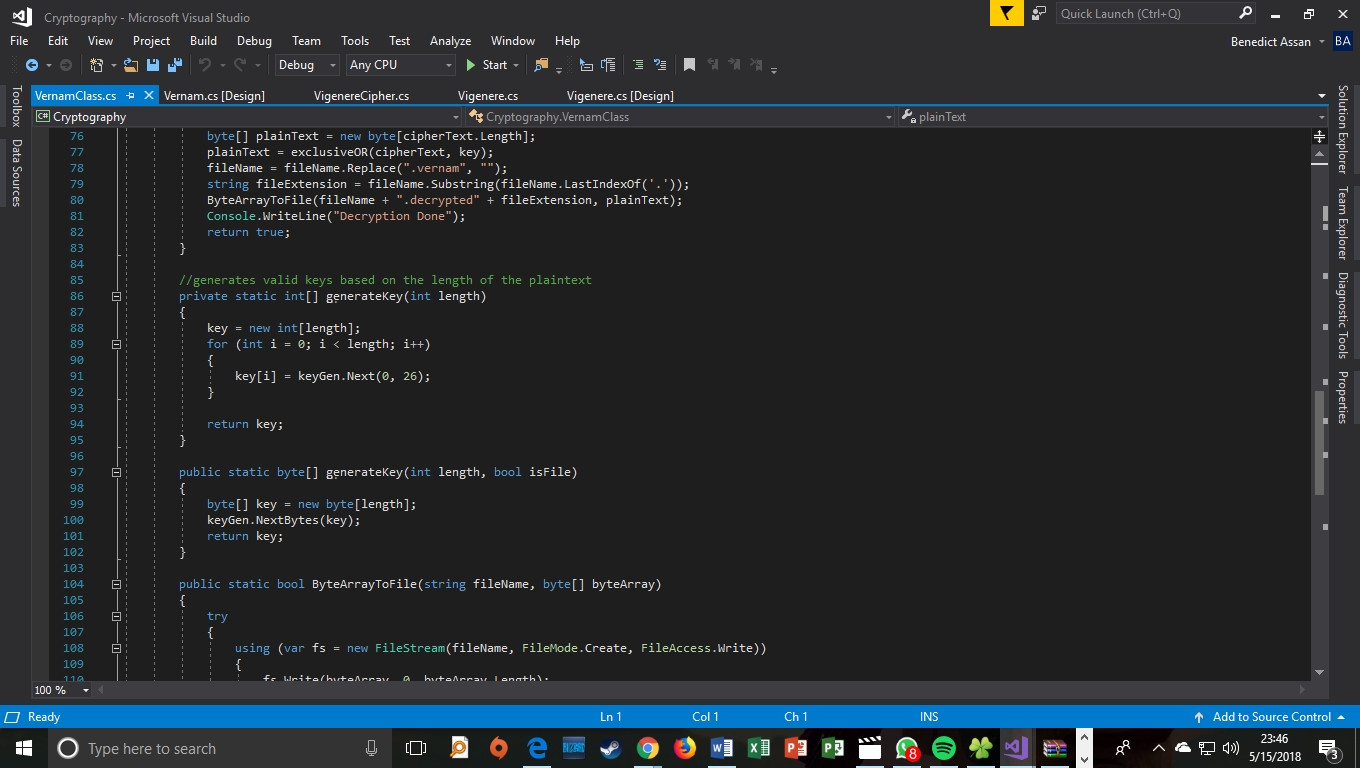
File encryption method:



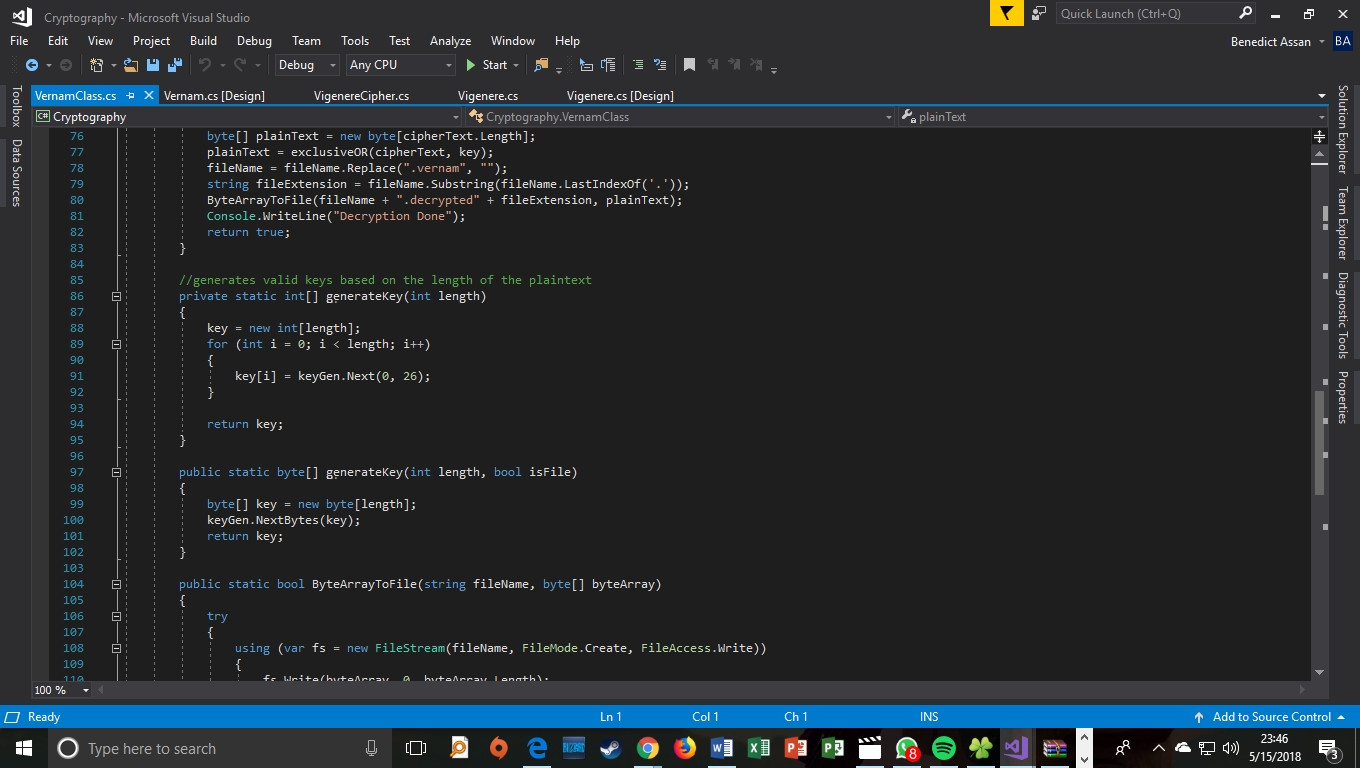
Exclusive OR method:



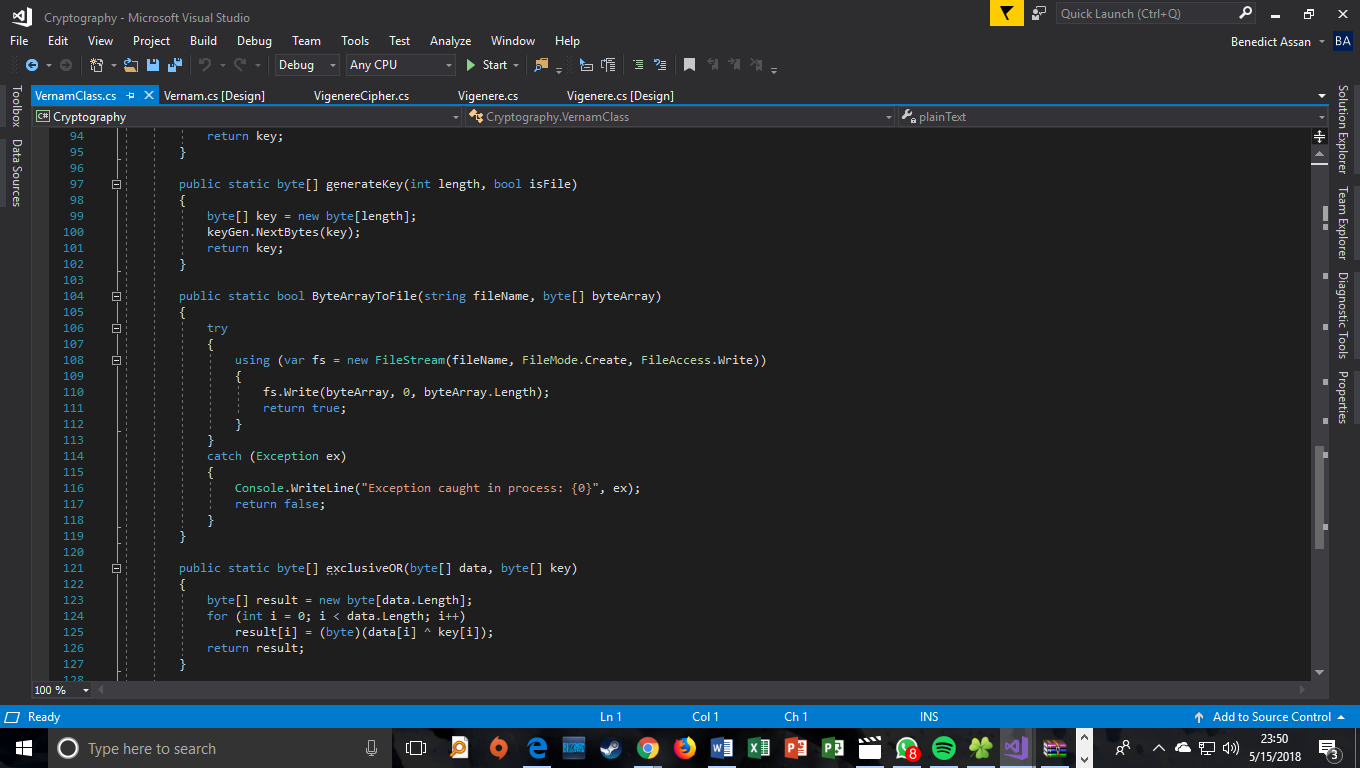
Key generation for plaintext:



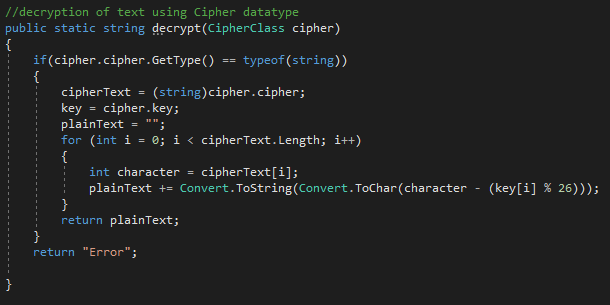
Key generation for file uploads:



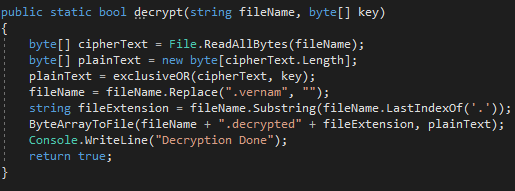
Method used to catch errors for invalid or corrupt files loaded into the system:



Decryption method used for plaintext:

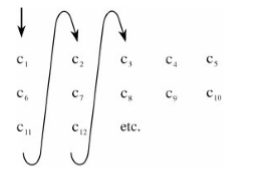


Decryption method used for files:

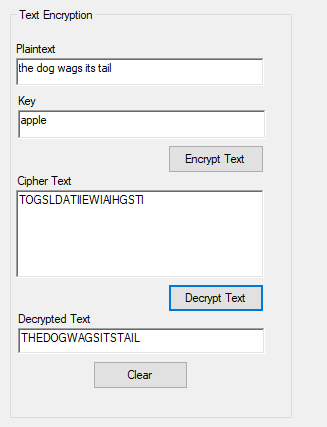


### Columnar Transposition

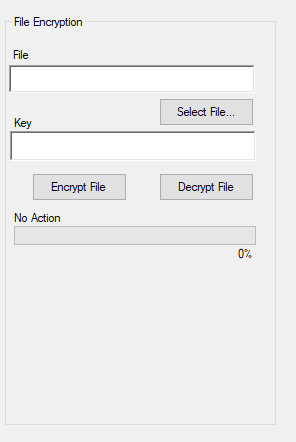
A transposition is an encryption method in which the letters of a plaintext messaged are rearranged, the sole purpose is to cause diffusion by widely spreading the information embedded in the plaintext or the key across the ciphertext. Columnar transposition in particular is a rearrangement of the letters or characters in the plaintext into columns with the ciphertext being formed by reading down the columns.



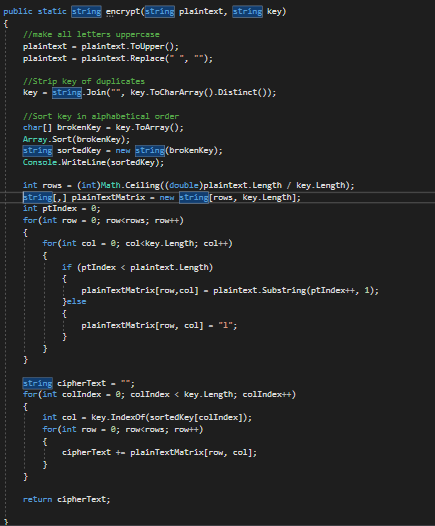
The program below performs a columnar transposition of plain text using ‘apple’ as a keyword.



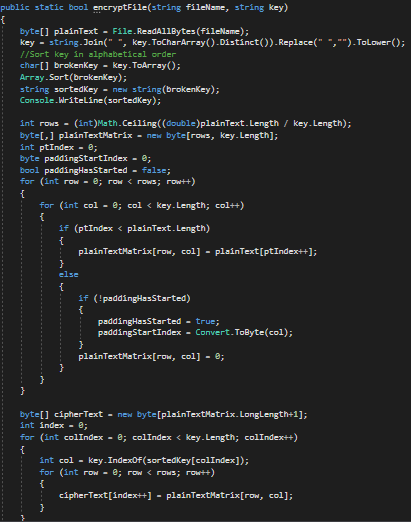
The program performs file encryption from the user uploading the file they wish to encrypt, the keyword they desire and pressing the ‘Encrypt file’ button which then creates the encrypted file in the user’s file system. If the user wishes to see the decrypted text, they can press the ‘Decrypt file’ button and access the decrypted file on their system.



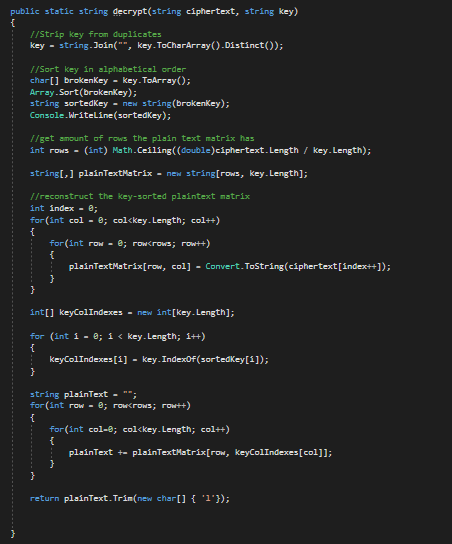
Plaintext Encryption Method:



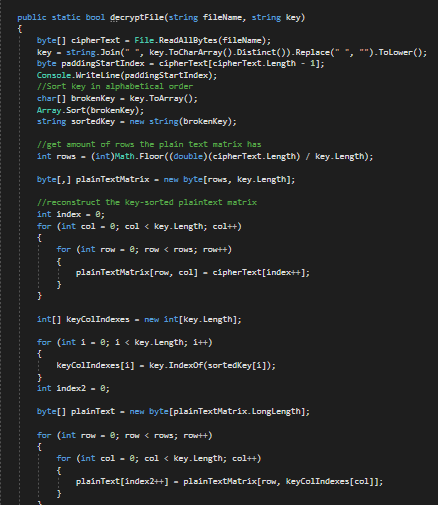
File encryption method:

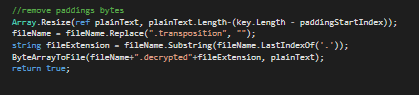


Plaintext decryption method:



File decryption method:





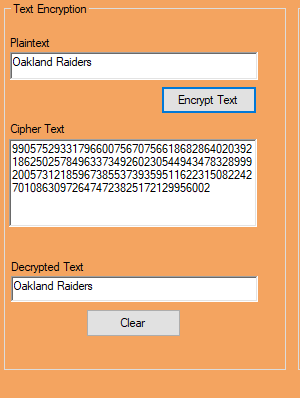
## RSA Algorithm (Advanced Encryption Standard)

The RSA algorithm is a public-key cryptosystem widely used for secure data transmission. It is also known as AES and is the standard specification for encrypting data as established by the U.S. National Institute of Standards and Technology in 2001. The algorithm involves four steps: key generation, key distribution, encryption and decryption. One of the basic principles behind the RSA is that it is practical to find three large positive integers – *e*, *d* and *n* such that with modular exponentiation for integer *m* and that even knowing the values of *e*, *n* or *m* make it challenging to find the value of *d*.

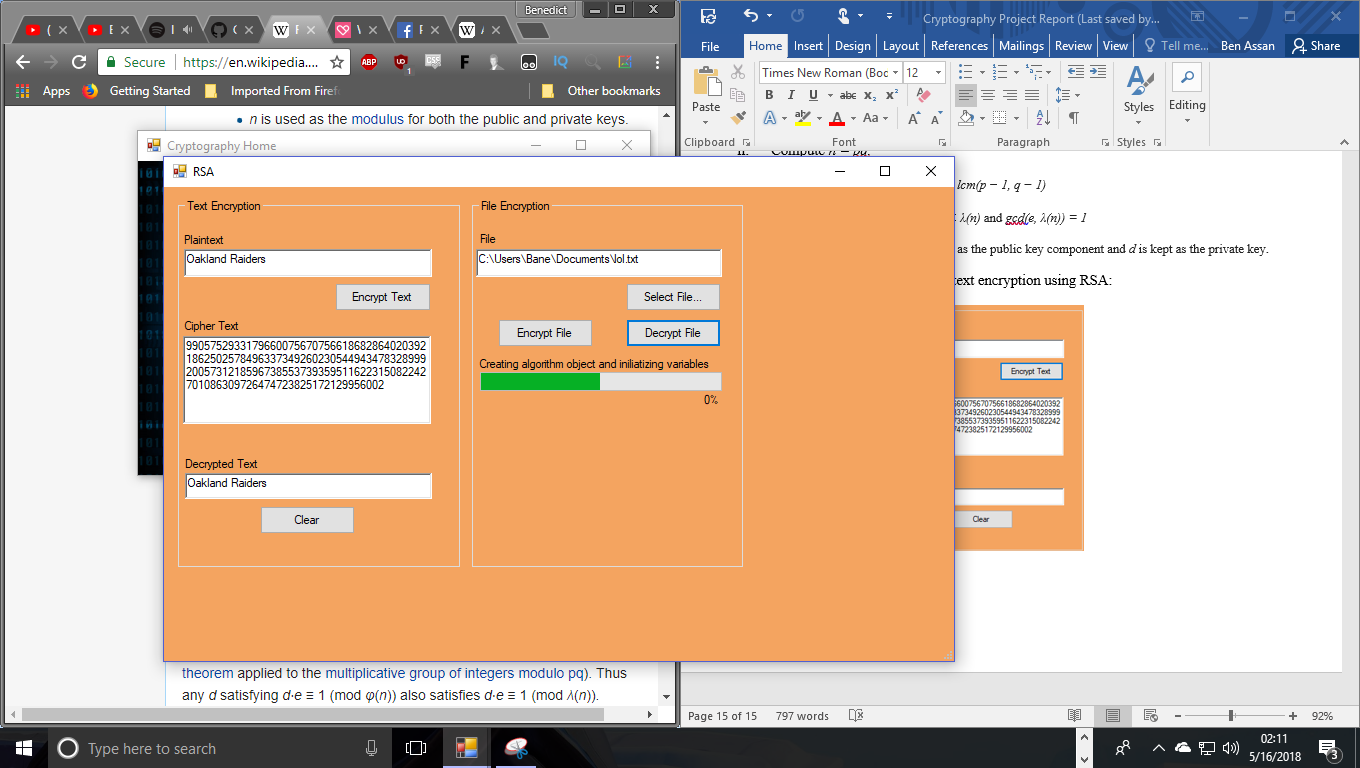
Keys are generated the following way:

* 1. Choose two distinct prime numbers *p* and *q*
  2. Compute *n = pq.*
  3. Compute  *λ(n) =*[*lcm*](https://en.wikipedia.org/wiki/Least_common_multiple)*(λ(p), λ(q)) = lcm(p − 1, q − 1)*
  4. Select an integer *e* such that *1 < e < λ(n)* and *gcd(e, λ(n)) = 1*
  5. Determine d as *d* = *e-1. e* is released as the public key component and *d* is kept as the private key.

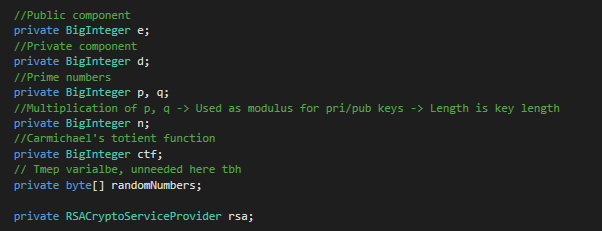
The diagram displayed below shows plaintext encryption using RSA:



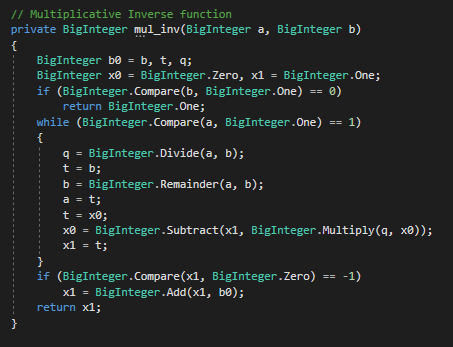
The diagram below shows the process of file encryption. The encrypted and decrypted files are saved onto the user’s file directory:



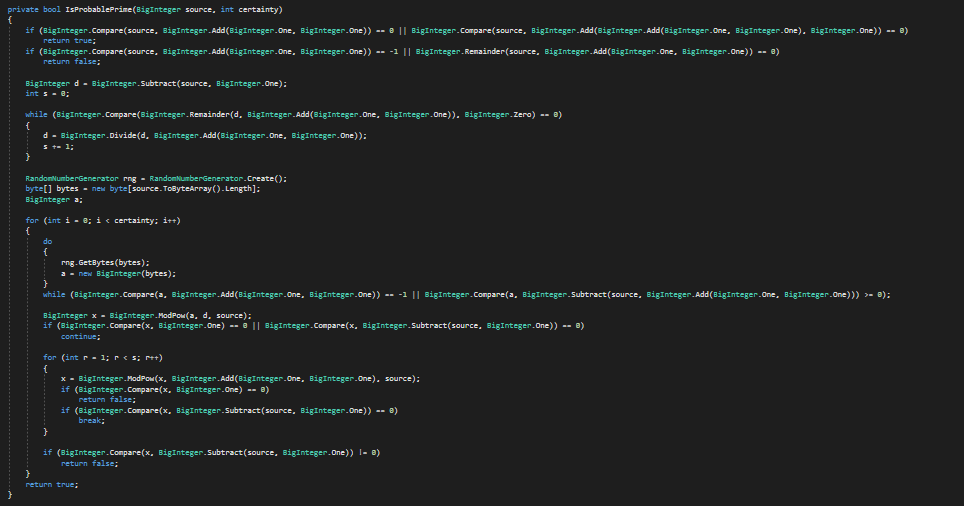
Global variables:



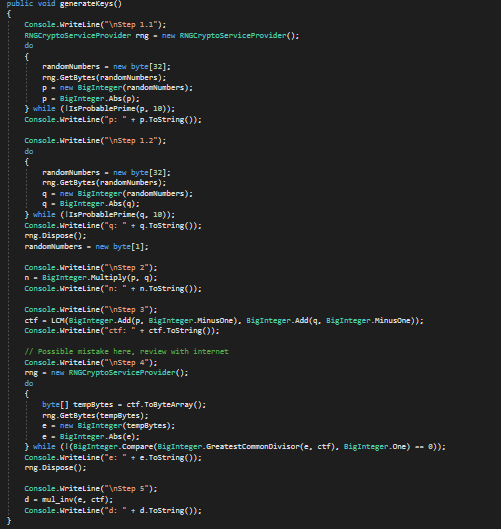
Multiplicative inverse function:



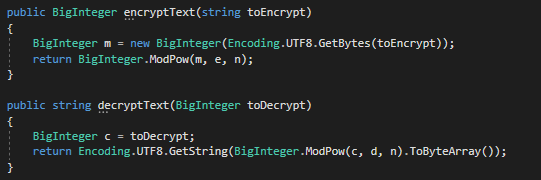
Method for determining a probable prime number:



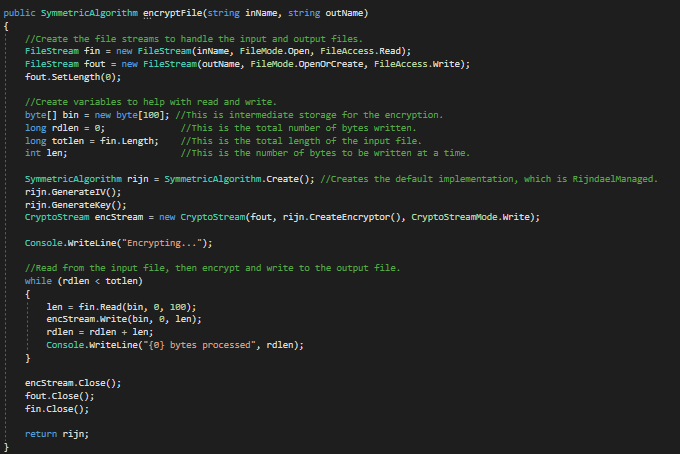
Key generation method:



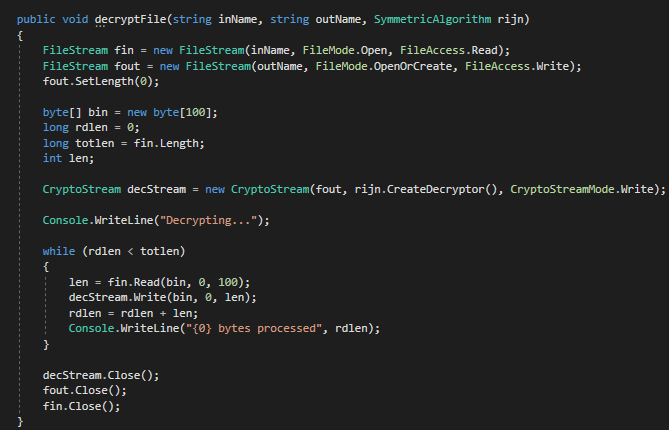
Plaintext encryption and decryption methods:



File encryption method:



File decryption method:



Least common multiple (LCM) method:

