Vigenere Cipher

# Prasidh aggarwal, 160907346, ece-d

## *abstract: it is shown with the help of a code blocks simulation that a series of caesar ciphers can be used to encrypt and decrypt the data, commonly referred to as the vigenere cipher.*

### Introduction: In a Caesar cipher, each letter of the alphabet is shifted along some number of places. For example, in a Caesar cipher of shift 3, A would become D, B would become E, Y would become B and so on. The Vigenere cipher has several Caesar ciphers in sequence with different shift values. To encrypt, a table of alphabets can be used, termed a *tabula recta*, *Vigenere square* or *Vigenere table*. It has the alphabet written out 26 times in different rows, each alphabet shifted cyclically to the left compared to the previous alphabet, corresponding to the 26 possible Caesar ciphers. At different points in the encryption process, the cipher uses a different alphabet from one of the rows. The alphabet used at each point depends on a repeating keyword.

**Algorithm:** Expressed mathematically, the encryption of the message at letter \*i\*, is equal to the alphabetic value of \*i\* in the plaintext plus the alphabetic value of the corresponding \*i\* in the key.

undefined

Decryption is the same process reversed, subtracting the key instead of adding to arrive back at the original, plaintext value.

### undefined

#### Simulation pseudo Code:

#include <iostream>

#include <string>

using namespace std;

class Vigenere

{

public:

string key;

Vigenere(string key)

{

for (int i = 0; i < key.size(); ++i)

{

if (key[i] >= 'A' && key[i] <= 'Z')

this->key += key[i];

else if (key[i] >= 'a' && key[i] <= 'z')

this->key += key[i] + 'A' - 'a';

}

}

string encrypt(string text)

{

string out;

for (int i = 0, j = 0; i < text.length(); ++i)

{

char c = text[i];

if (c >= 'a' && c <= 'z')

c += 'A' - 'a';

else if (c < 'A' || c > 'Z')

continue;

out += (c + key[j] - 2 \* 'A') % 26 + 'A';

j = (j + 1) % key.length();

}

return out;

}

string decrypt(string text)

{

string out;

for (int i = 0, j = 0; i < text.length(); ++i)

{

char c = text[i];

if (c >= 'a' && c <= 'z')

c += 'A' - 'a';

else if (c < 'A' || c > 'Z')

continue;

out += (c - key[j] + 26) % 26 + 'A';

j = (j + 1) % key.length();

}

return out;

}

};

int main()

{

Vigenere cipher("VIGENERECIPHER");

string original =

"Beware of the attack on the mainland!";

string encrypted = cipher.encrypt(original);

string decrypted = cipher.decrypt(encrypted);

cout << original << endl;

cout << "Encrypted: " << encrypted << endl;

cout << "Decrypted: " << decrypted << endl;

}

## Result:

During this project a very easy and non-tedious way of encrypting simple messages, and also decrypting them was found. It is not the best cipher there is, even kind of outdates these days. But it can be used in day to day life with easy as there is not much calculations involved.