**Cipher Research**

**Caesar**

The advantage of the Caesar Cipher is it’s simplicity. It’s a special case of the Shift Cipher when the key=3key=3. So the fact that is easy to implement it is the main advantage.

But even if the opponent trying to decipher the message doesn’t know that it was encrypted using Caesar Cipher but notices that it uses some kind of Shift Cipher, the key space of all possible keys has the size of only the size of the alphabet (let’s consider 26, for example). It is a very small size. A brute force attack trying all the 26 keys would break this crypto system easily.

In the special case of the Caesar Cipher, just doing:

xi−3(mod26)xi−3(mod26) ∀i∈{0,1,2,…,25}∀i∈{0,1,2,…,25}

wherewhere x0=a,x1=b,…,x25=zx0=a,x1=b,…,x25=z

Would give the opponent the plain text. So the disadvantage is the fact that is fairly easy to break it once you notice that it is being used.

Caesar Cipher is **not a secure** cryptosystem because there are only 26 possible keys to try out. An attacker can carry out an exhaustive key search with available limited computing resources.

**Vigenere**

Vigenere Cipher was designed by tweaking the standard Caesar cipher to reduce the effectiveness of cryptanalysis on the ciphertext and make a cryptosystem more robust. It is significantly **more secure than a regular Caesar Cipher**.

Vigenere cipher is a kind of substitution cipher that employs polyalphabetic substitutions. It is stronger than Caesar cipher as it uses series of interwoven Caesar ciphers.

But, it is a very old and practically useless cipher. Major disadvantage of Vigenere cipher is repeating nature of its keys. If a cryptanalyst correctly guesses the key's length, the cipher text can be treated as interwoven [Caesar ciphers](https://en.m.wikipedia.org/wiki/Caesar_cipher), which can easily be broken individually. The [Kasiski examination](https://en.m.wikipedia.org/wiki/Vigen%C3%A8re_cipher" \l "Kasiski_examination" \o "en.m.wikipedia.org" \t "_blank) and [Friedman test](https://en.m.wikipedia.org/wiki/Vigen%C3%A8re_cipher#Friedman_test) can help to determine the key length.

**Playfair**

It is also a substitution cipher and is difficult to break compared to the simple substitution cipher. As in case of substitution cipher, cryptanalysis is possible on the Playfair cipher as well, however it would be against 625 possible pairs of letters (25x25 alphabets) instead of 26 different possible alphabets.

The Playfair cipher was used mainly to protect important, yet non-critical secrets, as it is quick to use and requires no special equipment.

* It is significantly harder to break since the frequency analysis technique used to break simple substitution **ciphers** is difficult but still can be used on (25\*25) = 625 digraphs rather than 25 monographs which is difficult.
* Frequency analysis thus requires more **cipher** text to crack the encryption.

The symmetric **cryptography** through the **Playfair cipher** program technique can be easily cracked and the amount of data encryption and decryption will be less. It cannot be used for the transmission of a huge amount of data which is one of the most **disadvantage** factors of the technique.

**Atbash**

**Discussion**  
The Atbash Cipher is a very weak substitution cipher, since there is no secret key behind generating the ciphertext alphabet to perform the encryption. Thus, given a piece of ciphertext, known to have been enciphered using the Atbash Cipher, anyone who intercepts the message can easily decipher it to retrieve what was meant to be concealed.  
  
Despite this, it provides a very quick and easy way to conceal messages from an onlooker and can be used successfully to encipher messages of not great importance.  
  
The one security measure that does exist for the Atbash Cipher is to use different plaintext alphabets. For example using a plaintext alphabet with the ten digits attached at the end (ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789), or one with the most common punctuation marks in it. Both these alphabets add a bit more security to the cipher, but as we shall see, these are methods that can be used for every cipher.

// Java code to implement Vigenere Cipher

class GFG

{

// This function generates the key in

// a cyclic manner until it's length isi'nt

// equal to the length of original text

static String generateKey(String str, String key)

{

int x = str.length();

for (int i = 0; ; i++)

{

if (x == i)

i = 0;

if (key.length() == str.length())

break;

key+=(key.charAt(i));

}

return key;

}

// This function returns the encrypted text

// generated with the help of the key

static String cipherText(String str, String key)

{

String cipher\_text="";

for (int i = 0; i < str.length(); i++)

{

// converting in range 0-25

int x = (str.charAt(i) + key.charAt(i)) %26;

// convert into alphabets(ASCII)

x += 'A';

cipher\_text+=(char)(x);

}

return cipher\_text;

}

// This function decrypts the encrypted text

// and returns the original text

static String originalText(String cipher\_text, String key)

{

String orig\_text="";

for (int i = 0 ; i < cipher\_text.length() &&

i < key.length(); i++)

{

// converting in range 0-25

int x = (cipher\_text.charAt(i) -

key.charAt(i) + 26) %26;

// convert into alphabets(ASCII)

x += 'A';

orig\_text+=(char)(x);

}

return orig\_text;

}

// Driver code

public static void main(String[] args)

{

String str = "GEEKSFORGEEKS";

String keyword = "AYUSH";

String key = generateKey(str, keyword);

String cipher\_text = cipherText(str, key);

System.out.println("Ciphertext : "

+ cipher\_text + "\n");

System.out.println("Original/Decrypted Text : "

+ originalText(cipher\_text, key));

}

}

// This code has been contributed by 29AjayKumar