**LAB # 2**

**POLYALPHABETIC SUBSTITUTION CIPHER**

* A poly-alphabetic cipher is any cipher based on substitution techniques, using multiple substitution of alphabets.
* A letter in the plain text is not always replaced by the same letter in the cipher text
* We can use more than one substitution for the same letter

The selection of the could be selected from any of the three ways:

1. Sequential way
2. Keyword coded
3. Auto encoded

**Characteristics:**

* Increased Security as compare to mono-alphabetic ciphers.
* More resistant in frequency analysis because each letter can be encrypted in multiple ways.

**TYPES OF POLYALPHABETIC SUBSTITUTION CIPHER:**

1. Vigenere Cipher
2. Beaufort Cipher
3. Autokey Cipher
4. Enigma Cipher
5. **VIGENERE CIPHER:**

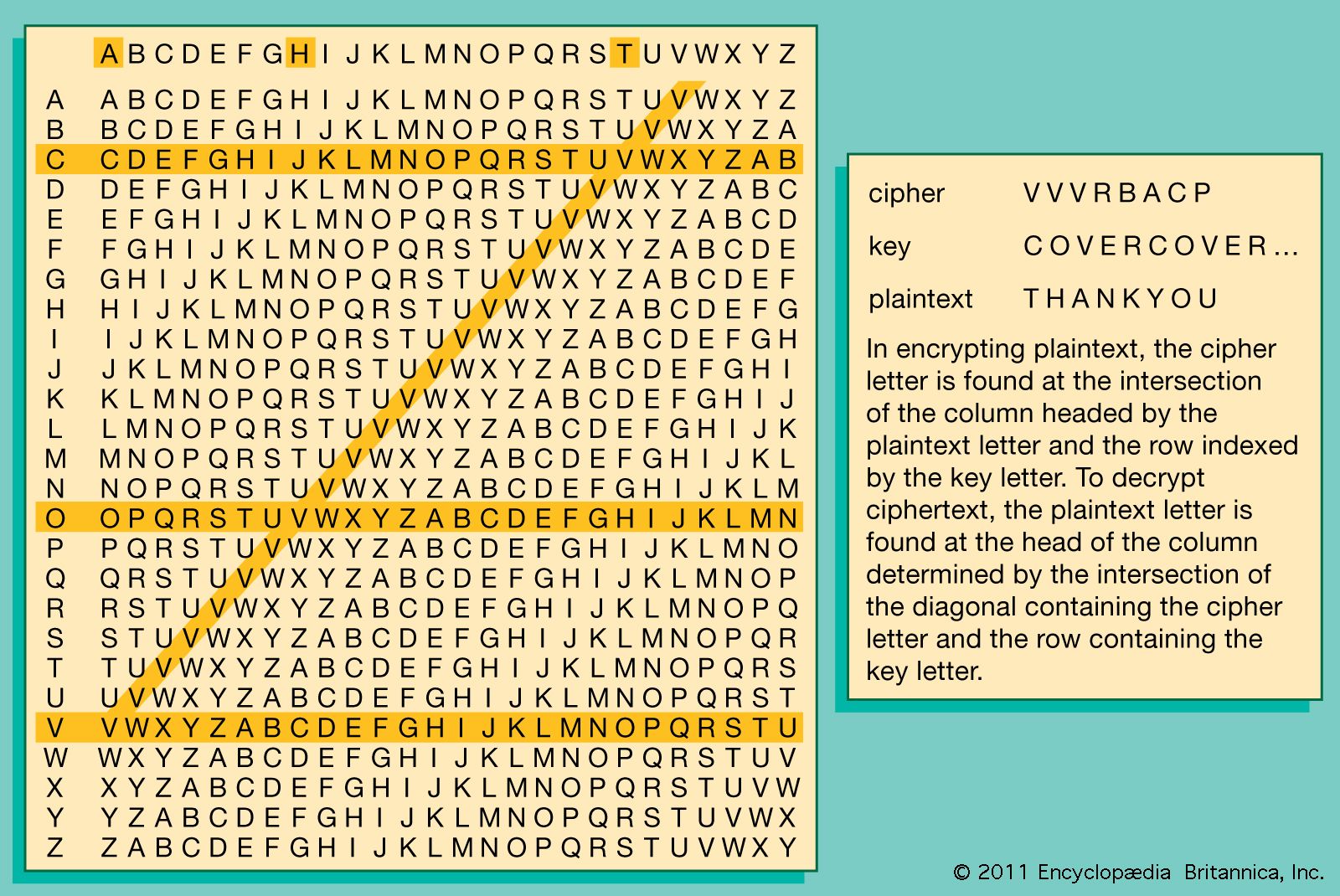
The Vigenere Cipher was first describe in 1953 by Giovan Battista Bellaso and it is used to encrypting and decrypting the text. It is easy to understand and implement.

It uses a vigenere table or vigenere square for encryption and decryption of the text. The vigenere table is also called the tabula racta.

It uses the series of interwovan caesar cipher.

**Characteristics:**

* The table consists of the 26 alphabets written out 26 times in different rows, each alphabet shift cyclically to the left compare to the previous alphabet, corresponding to the 26 possible caesar cipher.
* At different point in the encryption process, cipher uses a different alphabet from one of the rows.
* The alphabet uses in different point based on the repeating keywords.



**Method 1:**

**IMPLEMENTATION:**

**HTML:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Vigenère Cipher</title>

</head>

<body>

<div>

<div>

<h1>VIGENÈRE CIPHER</h1>

</div>

<div>

<label for="plain\_text">Input:</label>

<input type="text" name="encode" id="plain\_text" placeholder="Enter Plain Text">

<button id="encode" style="cursor: pointer;">Encode</button>

<p>Output is: <span id="encode\_result"></span></p>

</div>

<div>

<label for="cipher\_text">Input:</label>

<input type="text" name="decode" id="cipher\_text" placeholder="Enter Cipher Text">

<button id="decode" style="cursor: pointer;">Decode</button>

<p>Output is: <span id="decode\_result"></span></p>

</div>

</div>

<script src="script.js"></script>

</body>

</html>

**JAVASCRIPT:**

class VigenereCipher {

constructor() {

this.plain\_text = document.getElementById("plain\_text");

this.cipher\_text = document.getElementById("cipher\_text");

this.encode\_button = document.getElementById("encode");

this.decode\_button = document.getElementById("decode");

this.encode\_result = document.getElementById("encode\_result");

this.decode\_result = document.getElementById("decode\_result");

this.alphabet = 'abcdefghijklmnopqrstuvwxyz';

this.matrix = [];

this.key = '';

this.get\_26\_by\_26\_matrix();

this.encrypt();

this.decrypt();

}

get\_26\_by\_26\_matrix() {

for (let i = 0; i < this.alphabet.length; i++) {

let row = [];

let j = i;

while (row.length != 26) {

row.push(this.alphabet[j % 26]);

j++;

}

this.matrix.push(row);

}

}

set\_key(key, length) {

while (key.length < length) {

key += key;

}

if (this.plain\_text.value.indexOf(" ") != -1) {

let space = this.plain\_text.value.indexOf(" ");

key = key.slice(0, space) + " " + key.slice(space);

}

this.key = key.slice(0, length);

}

encrypt() {

this.encode\_button.addEventListener('click', () => {

let key = prompt("Enter the key:");

this.set\_key(key, this.plain\_text.value.length);

let cipher\_text = "";

for (let i = 0; i < this.plain\_text.value.length; i++) {

if (this.plain\_text.value[i] === ' ') {

cipher\_text += ' ';

} else {

let row = this.alphabet.indexOf(this.key[i]);

let column = this.alphabet.indexOf(this.plain\_text.value[i]);

cipher\_text += this.matrix[row][column];

}

}

this.encode\_result.innerText = cipher\_text;

});

}

decrypt() {

this.decode\_button.addEventListener('click', () => {

let key = prompt("Enter the key:");

this.set\_key(key, this.cipher\_text.value.length);

let plain\_text = "";

for (let i = 0; i < this.cipher\_text.value.length; i++) {

if (this.cipher\_text.value[i] === ' ') {

plain\_text += ' ';

} else {

let row = this.alphabet.indexOf(this.key[i]);

let column = this.matrix[row].indexOf(this.cipher\_text.value[i]);

plain\_text += this.alphabet[column];

}

}

this.decode\_result.innerText = plain\_text;

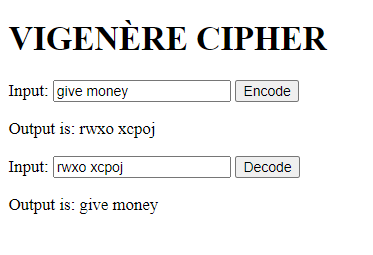
});

}

}

let obj = new VigenereCipher();

**OUTPUT/UI :**



**METHOD 2:**

**Encryption: (convert plain text into cipher text)**

E = (P,K) = (P+K)%26

**Decryption:(convert cipher text into plain text )**

D = (C,K) = (C+K)%26

**Example:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Plain text** | **G** | **I** | **V** | **E** | **M** | **O** | **N** | **E** | **Y** |
| **Key** | **L** | **O** | **C** | **K** | **L** | **O** | **C** | **K** | **L** |
| **Plain text value** | **6** | **8** | **21** | **4** | **12** | **14** | **13** | **4** | **24** |
| **Key value** | **11** | **14** | **2** | **10** | **11** | **14** | **2** | **10** | **11** |
| **Encryption** | **17** | **22** | **23** | **14** | **23** | **2** | **15** | **14** | **9** |
| **Cipher text** | **R** | **W** | **X** | **O** | **X** | **C** | **P** | **O** | **T** |

**IMPLEMENTATION:**

**HTML:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Vigenere Cipher</title>

</head>

<body>

<div>

<div>

<h1>Vigenere Cipher</h1>

</div>

<div>

<label for="plain\_text">Input:</label>

<input type="text" name="encode" id="plain\_text" placeholder="Enter Plain Text">

<button id="encode" style="cursor: pointer;">Encode</button>

<p>Output is: <span id="encode\_result"></span></p>

</div>

<div>

<label for="cipher\_text">Input:</label>

<input type="text" name="decode" id="cipher\_text" placeholder="Enter Cipher Text">

<button id="decode" style="cursor: pointer;">Decode</button>

<p>Output is: <span id="decode\_result"></span></p>

</div>

</div>

<script src="script.js"></script>

</body>

</html>

**Javascript:**

class Vigenere\_Cipher {

constructor() {

this.plain\_text = document.getElementById("plain\_text");

this.cipher\_text = document.getElementById("cipher\_text");

this.encode\_button = document.getElementById("encode");

this.decode\_button = document.getElementById("decode");

this.encode\_result = document.getElementById("encode\_result");

this.decode\_result = document.getElementById("decode\_result");

this.encrypt();

this.decrypt();

}

#generate\_key = (message, key) => {

key = key.split("");

if (message.length == key.length) {

return key.join("");

} else {

let temporary\_key = key;

for (let i = 0; key.length < message.length; i++) {

key.push(temporary\_key[i % temporary\_key.length]);

}

return key.join("");

}

}

encrypt = () => {

this.encode\_button.addEventListener('click', () => {

let plain\_text = this.plain\_text.value;

let key = prompt("Enter the Key");

key = this.#generate\_key(plain\_text, key);

let cipher\_text = "";

for (let i = 0; i < plain\_text.length; i++) {

let x = (plain\_text.charCodeAt(i) + key.charCodeAt(i)) % 26;

x += 'A'.charCodeAt(0);

cipher\_text += String.fromCharCode(x);

}

this.encode\_result.innerText = cipher\_text;

});

}

decrypt = () => {

this.decode\_button.addEventListener('click', () => {

let cipher\_text = this.cipher\_text.value;

let key = prompt("Enter the Key");

key = this.#generate\_key(cipher\_text, key);

let plain\_text = "";

for (let i = 0; i < cipher\_text.length; i++) {

let x = (cipher\_text.charCodeAt(i) - key.charCodeAt(i) + 26) % 26;

x += 'A'.charCodeAt(0);

plain\_text += String.fromCharCode(x);

}

this.decode\_result.innerText = plain\_text;

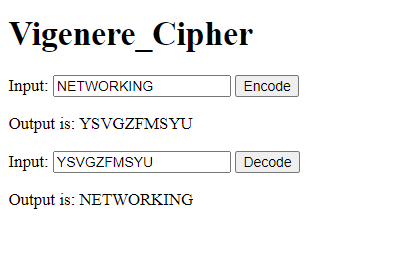
});

}

}

let obj = new Vigenere\_Cipher();

**OUTPUT/UI:**

****

1. **POLY-ALPHABETIC CIPHER:**

**IMPLEMENTATION:**

**HTML:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Polyalphabetic</title>

</head>

<body>

<div>

<div>

<h1>Polyalphabetic Cipher</h1>

</div>

<div>

<label for="plain\_text">Input:</label>

<input type="text" name="encode" id="plain\_text" placeholder="Enter Plain Text">

<button id="encode" style="cursor: pointer;">Encode</button>

<p>Output is: <span id="encode\_result"></span></p>

</div>

<div>

<label for="cipher\_text">Input:</label>

<input type="text" name="decode" id="cipher\_text" placeholder="Enter Cipher Text">

<button id="decode" style="cursor: pointer;">Decode</button>

<p>Output is: <span id="decode\_result"></span></p>

</div>

</div>

<script src="script.js"></script>

</body>

</html>  
  
**Javascript:**

const header = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z'];

const matrix = [

['a', 's', 'd', 'f', 'g', 'h', 'j', 'k', 'l', 'p', 'o', 'i', 'u', 'y', 't', 'r', 'e', 'w', 'q', 'z', 'x', 'c', 'v', 'b', 'n', 'm'],

['q', 'a', 'z', 'w', 's', 'x', 'e', 'd', 'c', 'r', 'f', 'v', 't', 'g', 'b', 'y', 'h', 'n', 'u', 'p', 'j', 'm', 'i', 'k', 'o', 'l'],

['z', 'x', 'c', 'v', 'b', 'n', 'm', 'k', 'i', 'o', 'p', 'l', 'u', 'j', 'h', 'y', 't', 'g', 'f', 'r', 'e', 'd', 's', 'w', 'q', 'a']

];

const getMatrixIndex = (i) => (i < 3) ? (i + 3) % 3 : i % 3;

const cipherText = (plainText) => {

let cipher\_text = "";

for (let i = 0; i < plainText.length; i++) {

let index = header.indexOf(plainText[i]);

if (index !== -1) {

let get\_matrix = getMatrixIndex(i);

cipher\_text += matrix[get\_matrix][index];

} else {

cipher\_text += plainText[i];

}

}

return cipher\_text;

};

const decipherText = (cipherText) => {

let plain\_text = "";

for (let i = 0; i < cipherText.length; i++) {

let get\_matrix = getMatrixIndex(i);

let index = matrix[get\_matrix].indexOf(cipherText[i]);

if (index !== -1) {

plain\_text += header[index];

} else {

plain\_text += cipherText[i];

}

}

return plain\_text;

};

document.getElementById('encode').addEventListener('click', () => {

const plainText = document.getElementById('plain\_text').value.toLowerCase();

const result = cipherText(plainText);

document.getElementById('encode\_result').textContent = result;

});

document.getElementById('decode').addEventListener('click', () => {

const cipherText = document.getElementById('cipher\_text').value.toLowerCase();

const result = decipherText(cipherText);

document.getElementById('decode\_result').textContent = result;

});

**OUTPUT/UI:**

****