Wheatstone-Playfair Cipher

Digraphic Square Substitution Cipher

Keane J. Moraes & Nadeem Ahamed

1. History

The Playfair Cipher was invented by Sir Charles Wheatstone (FRS) in 1854 for secrecy in telegraphy. It was the first cipher in history to encrypt pairs of letters at the same time.[1]. It is called the 'Playfair' system because it was Lyon Playfair who promoted its use. Lyon Playfair, the first Baron Playfair of St. Andrews, championed its use at the British Foreign Office. Once Wheatstone established that with proper training even school children could properly use the Playfair cipher, Britain would make the Playfair cipher their prominent tool to encrypt secret, but non-critical information in the battlefields.

2. Modern Usage

This was used by the British forces in the Second Boer War and in World War I. Other countries–Australia, Germany, and New Zealand–would use the Playfair cipher in the 1940's. When Lt. John F. Kennedy's PT-109 was sunk by a Japanese cruiser in the Solomon Islands, for instance, he made it to shore on Japanese-controlled Plum Pudding Island and was able to send an emergency message in Playfair from an Allied coast-watcher's hut to arrange the rescue of the survivors from his crew.[2] However, after the advent of computers and its use to break codes, the Playfair cipher was rarely used. The rise of sophisticated digital encryption schemes meant that the Playfair cipher would be finally put to rest.

3. Encryption and Decryption Algorithms

The encryption and decryption formulae differ but the code implementation is almost identical. The only major difference is the use of subtraction instead of addition.

3.1. Encryption

The Encryption algorithm is fragmented into 3 cases :

- Same Row
- Same Column
- General

One can make a unified formula for encryption but that will not be discussed here. The code implementation for the cases is discussed in tandem with the visual model. Before we move onto the cases, we must first discuss how to place the letters in the square. This is done through the Polybius Square encryption algorithm.

$$R_i = \left\lfloor \frac{i}{5} \right\rfloor + 1$$

$$C_i = i \mod 5 + 1$$

[1] Cohen, Fred. (1995). A Short History of Cryptography, Retrieved from all.net/edu/curr/ip/Chap2-1.html

[2] From Playfair Cipher - PBS.com

NOTE
We will use the custom keys method seen in the Polybius Square segue on alternate keys. This allows for a more robust encryption scheme.

3.1.1. Code Implementation

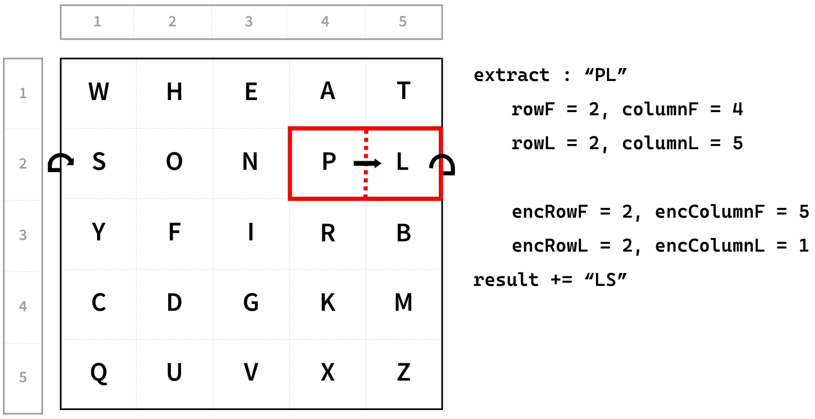
At first, 2 characters at a time are extracted. If the characters are not in the key (the modified English Alphabet) they are just tacked onto the result.

```
String extract = plainText.substring(i,i+2);

// STEP 1 : WEED OUT THE 'EXTRAS'
if(key.indexOf(extract.charAt(0)) == -1 || key.indexOf(extract.charAt(1)) == -1){
  result += extract;
  continue;
}// if statement - NOT IN KEY MATRIX
```

In the Polybius Square encryption algorithm, we did this per letter. Now we are doing this for a pair of letters. These are identical to code implementation of the formulae in Polybius.

```
// STEP 2 : EXTRACT THE ROWS AND COLUMNS OF BOTH THE CHARACTERS
rowF = key.indexOf(extract.charAt(0)) / 5;
rowL = key.indexOf(extract.charAt(1)) / 5;
columnF = key.indexOf(extract.charAt(0)) % 5;
columnL = key.indexOf(extract.charAt(1)) % 5;
```

3.1.2. Same Row



	1	2	3	4	5
1	W	H	E	A	T
2	S	O	N	P	L
3	Y	F	I	R	B
4	C	D	G	K	M
5	Q	U	V	X	Z

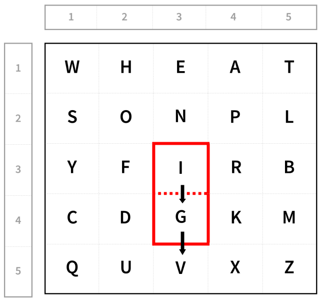
```
extract : "PL"
    rowF = 2, columnF = 4
    rowL = 2, columnL = 5

    encRowF = 2, encColumnF = 5
    encRowL = 2, encColumnL = 1
result += "LS"
```

When the two characters are located on the same row then, they are encrypted by shifting both of them to the right while wrapping if need be.

Code Implementation The code implementation is straightfoward - the encrypted characters have the same row of the plain text characters and their columns are added by 1 while within mod 5.

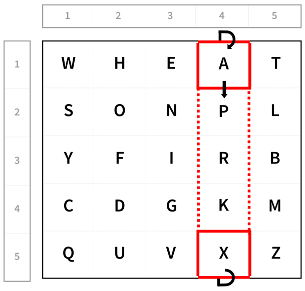
```
if (rowF == rowL) {
  encColumnF = (columnF + 1) % 5;
  encColumnL = (columnL + 1) % 5;
  encRowF = encRowL = rowL;
} // if statement - same ROW
```

3.1.3. Same Column



	1	2	3	4	5
1	W	H	E	A	T
2	S	O	N	P	L
3	Y	F	I	R	B
4	C	D	G	K	M
5	Q	U	V	X	Z

```
extract : “IG”
    rowF = 3, columnF = 3
    rowL = 4, columnL = 3

    encRowF = 4, encColumnF = 3
    encRowL = 5, encColumnL = 3
result += “GV”
```



	1	2	3	4	5
1	W	H	E	A	T
2	S	O	N	P	L
3	Y	F	I	R	B
4	C	D	G	K	M
5	Q	U	V	X	Z

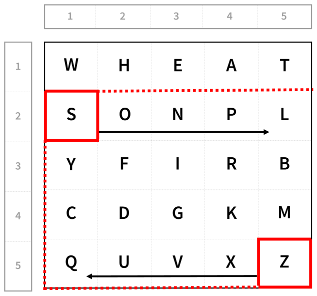
```
extract : “AX”
    rowF = 1, columnF = 4
    rowL = 5, columnL = 4

    encRowF = 2, encColumnF = 4
    encRowL = 1, encColumnL = 4
result += “PA”
```

When the two characters are located on the same column then, they are encrypted by shifting both of them to down while wrapping if need be.

Code Implementation The code implementation is straightfoward - the encrypted characters have the same column of the plain text characters and their rows are added by 1 while within $\mod 5$.

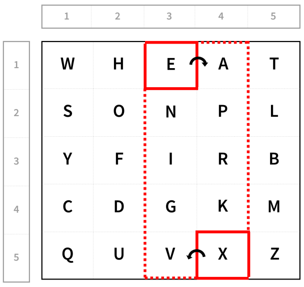
```
    else if (columnF == columnL) {
      encRowF = (rowF + 1) % 5;
      encRowL = (rowL + 1) % 5;
      encColumnF = encColumnL = columnF;
    } // if statement - same COLUMN
```

3.1.4. General Case



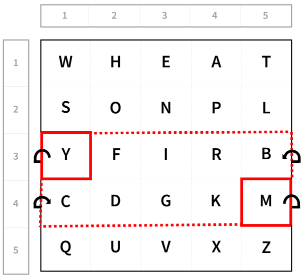
	1	2	3	4	5
1	W	H	E	A	T
2	S	O	N	P	L
3	Y	F	I	R	B
4	C	D	G	K	M
5	Q	U	V	X	Z

```
extract : “ZS”
    rowF = 5, columnF = 5
    rowL = 2, columnL = 1

    encRowF = 2, encColumnF = 5
    encRowL = 5, encColumnL = 1
result += “LQ”
```



	1	2	3	4	5
1	W	H	E	A	T
2	S	O	N	P	L
3	Y	F	I	R	B
4	C	D	G	K	M
5	Q	U	V	X	Z

```
extract : “XE”
    rowF = 5, columnF = 4
    rowL = 1, columnL = 3

    encRowF = 5, encColumnF = 3
    encRowL = 1, encColumnL = 4
result += “VA”
```



	1	2	3	4	5
1	W	H	E	A	T
2	S	O	N	P	L
3	Y	F	I	R	B
4	C	D	G	K	M
5	Q	U	V	X	Z

```
extract : “YM”
    rowF = 3, columnF = 1
    rowL = 4, columnL = 5

    encRowF = 3, encColumnF = 5
    encRowL = 4, encColumnL = 1
result += “BC”
```

In the general case, the rows of the letters stay the same but the columns get swapped.

Code Implementation The code implementation is trivial - the encrypted characters have the same row of the plain text characters and their column are swapped around.

```
    else {
      encRowF = rowF;
      encColumnF = columnL;
      encRowL = rowL;
      encColumnL = columnF;
    } // else - REST OF THE CASES
```

3.2. Decryption

Decryption is almost the same as encryption. Instead of adding to the column or row we subtract instead and ensure that we wrap around by doing $\mod 5$. intind