



# LAYFPYR

## PLAY FAIR

It made for security 429CSC project

Done by:

Nada Al-Otaibi , Noura Al-Degaither , Renad Al-Oaidi

# CONTENTS

|  |    |
|--|----|
| 1. INTRODUCTION .....                              | 2  |
| 2. WHAT IS A CRYPTO-SYSTEM AND HOW IT WORKS? ..... | 2  |
| 3.1 WHAT IS PLAYFAIR AND HOW IT WORKS? .....       | 2  |
| 3.2 WHO INVENTED PLAYFAIR? .....                   | 3  |
| 3.3 ADVANTEGES AND DISADVANTEGES:.....             | 3  |
| 4. CODE.....                                       | 4  |
| 5. RUN TIME RESULTS (SCREENSHOTS) .....            | 11 |
| 6. REFERENCES:.....                                | 14 |

## 1. Introduction

In the world of information security cryptography is considered a very prevalent tool. It is essential for either keeping communications secrecy over open channels or for proving the authenticity of an incoming message. In actual fact, the whole range of applicability is very wide, and it would not be possible to give a complete list of functionalities that can be achieved through cryptography [1]. When we talk about cryptography there a huge number of methods that can be used. However, for this project the cryptography method we chose to work on is Playfair cipher, which is a slightly classical method, we are going to make a GUI based program using Java programming language, we chose Java programming language because we are familiar with it. In this report firstly we are going to present some information about the Playfair cipher, the advantages, and disadvantages of this method, also we are going to present the important parts of the code along with the runtime results.

## 2. What is a crypto-system and how it works?

Cryptographic calculations assume a significant function in the security engineering of any correspondence organization. One kind of these calculations depends on the symmetric key, which encrypts and decrypts information utilizing one key. There are fundamentally two different ways to make a more grounded figure: the stream figure, where the encryption rule is created relying upon a plantent image's situation in the flood of plantent images, and the square code, which scrambles a few plantent images on the double in a square [2].

### 3.1 What is Playfair and how it works?

Remarkable among different acknowledged early square codes is the Playfair machine. Diverged from similarly advanced statistics encryption processes, which have complex computationnel avances, Playfair figures are commonly less stunning. From the computationnel and hardware factor of view, greater eccentric counts require greater strength use, which makes them less enticing for use in remote gadgets, as an example, telephones, some distance off the sensor, and many others. Conversely, Playfair parent, being almost simpler than their moreover confounding accomplices, are low power utilization figuring's and in like way are proper for statistics protection in far off packages [3].

The Playfair determine became the vital code to encode sets of letters in cryptologic history. Wheatstone constructed up the code for the puzzle in media transmission, yet it passes on the call of his buddy Master Playfair, first Aristocrate Playfair of St. Andrews, who contemporary its usage.

### 3.2 Who invented Playfair?

The scheme was invented in 1854 by Charles Wheatstone but bears the name of Lord Playfair for promoting its use.

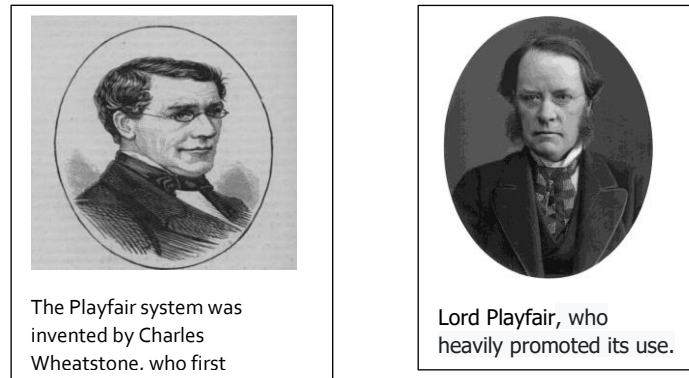


Figure 1

### 3.3 Advantages and disadvantages:

It is in general sense harder to break since the repeat examination's technique used to break fundamental substitution figures is inconvenient yet can be used on 625 digraphs rather than 25 monographs which is problematic [4].

In the other hand, a weakness is the fact that a digraph in the ciphertext and its reverse will have corresponding plaintexts (also ciphertext will correspond to plaintext, i.e. the substitution is self-inverse). This is easily exploited with the help of frequency analysis, if the language of the plaintext is known.

Also, the Playfair cipher is a symmetric cipher thus the same key is used for both encryption and decryption.

## 4. Code

In this section we will present the most important methods in our Playfair cipher program along with an explanation for each of them.

### prepareInputText Method:

```
private char[][] prepareInputText(String inputText)//preparing inputtext to be encrypted/decrypted
{
    inputText = inputText.toUpperCase();
    inputText = inputText.replaceAll("\\s", ""); //remove all whitespaces
    String inputTextN = "";
    int b = 0;
    int c = 0;
    for(int i=0; i<inputText.length(); i++) { //preparing the input text if there were two similar lett
        if(i==0) {
            if(inputText.charAt(b) == 'J') {
                inputTextN = inputTextN + 'I';
            }
            else {
                inputTextN = inputTextN + inputText.charAt(b);
            }
            b++;
        }
        else if(i%2!=0 && inputText.charAt(b)==inputText.charAt(c) && inputTextN.length()%2!=0) {
            if(inputText.charAt(b) == 'J') {
                inputTextN = inputTextN + 'X' + 'I';
            }
            else {
                inputTextN = inputTextN + 'X' + inputText.charAt(b);
            }
            b++;
            c++;
        }
        else{
            if(inputText.charAt(b) == 'J') {
                inputTextN = inputTextN + 'I';
            }
            else {
                inputTextN = inputTextN + inputText.charAt(b);
            }
            b++;
            c++;
        }
    }
    ///////////////////////////////////////////////////
    if(inputTextN.length()%2 != 0)//putting all inputtext characters in matrix
    {
        inputTextN = inputTextN + 'X';
    }

    char[][] matr = new char[inputTextN.length()/2][2];
    int k = 0;
    for(int i=0; i<inputTextN.length()/2; i++)
    {
        for(int j=0; j<2; j++)
        {
            matr[i][j] = inputTextN.charAt(k);
            k++;
        }
    }
    return matr;
}
```

prepareInputText method which will be preparing the input text according to Playfair cipher algorithm to get it decrypted or encrypted. First, the method will be taking the input string, removing all white spaces, and converting it to uppercase. Initializing a new input text String which will be the input text after the editing, since the input should be a matrix of (n/2x2) we need to make sure that no similar letters are together in one pair(row). We will be using 'X' to separate them. In addition, we need to make sure that every 'J' is converted to 'I' since the Playfair table is 5x5 and 'I' = 'I' / 'J'. Finally, making sure that the length of the input text is even and if it is not, we append 'X' in the end. By the end of the method, the new edited input text is assigned to a (n/2x2) matrix.

### prepareKey Method:

```

////////////////////////////////////
public char[] prepareKey(char[] key)//preparing the key
{
    List<Character> Key2 = new ArrayList<Character>();
    for(int i = 0; i < key.length; i++)
    {
        if(!Key2.contains(key[i]))
        {
            Key2.add(key[i]);
        }
    }
    if(Key2.contains('J'))// I = I/J
    {
        int i = Key2.indexOf('J');
        Key2.remove(i);
        Key2.add(i, 'I');
    }
    char [] finalKey = ListToCharArray(Key2);
    return finalKey;
}
////////////////////////////////////

```

prepareKey method goal is to prepare the key that will be used to decrypt or encrypt the input text according to Playfair cipher algorithm. First, a list of characters typed key2 is initialized, this list will contain the key after the changes, so values of key will be assigned to it, however we make sure that the letters are not repeated, if a letter is already in key2 then we skip it. When the assigning is done, we will make sure if the key is containing 'J', if it does, we replace it with 'I'. Finally, we convert the list key to array of characters using a helping method to do this.

### prepareTable Method:

```
////////////////////////////////////  
////////////////////////////////////  
public char[][] prepareTable(char[] key) //preparing the playfair table/mat  
{  
  
    char[][] matrix = new char[5][5];  
    int k = 0;  
    |  
    for(int i=0; i < key.length; i++)  
    {  
        Letters.remove(Letters.indexOf(key[i]));  
        Letters.add(0, key[i]);  
    }  
  
    for(int i = 0 ; i < 5; i++)  
    {  
        for(int j = 0; j < 5 ; j++)  
        {  
            matrix[i][j] = Letters.get(k);  
            k++;  
        }  
    }  
    return matrix;  
}  
////////////////////////////////////
```

prepareTable method goal is to prepare the table that will be used for the encryption or decryption of the input text according to Playfair cipher algorithm. First, the list of letters will be sorted in compliance to the prepared key, a matrix of size 5x5 is initialized and every character in the letters list will be assigned to it accordingly.

## encryptText & encryptDi Methods:

```
////////////////////////////////////
private String encryptText(String key, String inputText)//if the inputtext wa
{
    char[][] matr = prepareInputText(inputText);
    Table playfairB = new Table(key);
    String encryptedText = "";

    for(int i = 0; i < matr.length; i++)
    {
        encryptedText = encryptedText + playfairB.encryptDi(matr[i]);
    }

    return encryptedText;
}
////////////////////////////////////

////////////////////////////////////
public String encryptDi(char[] matr)
{
    //matrix => nX2
    char[] output = new char[2];
    Integer[] coordsOfChar0 = idx.get(new Character(matr[0]));
    Integer[] coordsOfChar1 = idx.get(new Character(matr[1]));

    int X0 = coordsOfChar0[0].intValue();
    int X1 = coordsOfChar1[0].intValue();
    int Y0 = coordsOfChar0[1].intValue();
    int Y1 = coordsOfChar1[1].intValue();

    if(X0 != X1 && Y0 != Y1)//both different => taking the Intersection
    {
        output[0] = this.cipherMatrix[X0][Y1];
        output[1] = this.cipherMatrix[X1][Y0];
    }
    else if(X0 != X1 && Y0 == Y1)//same column => shift down
    {
        output[0] = this.cipherMatrix[getEncryptCoord(X0)][Y0];
        output[1] = this.cipherMatrix[getEncryptCoord(X1)][Y1];
    }
    else if(X0 == X1 && Y0 != Y1)//same row => shift right
    {
        output[0] = this.cipherMatrix[X0][getEncryptCoord(Y0)];
        output[1] = this.cipherMatrix[X1][getEncryptCoord(Y1)];
    }
    String strOutput = ArrayToString(output);
    return strOutput;
}
////////////////////////////////////
```



### Helping Method:

```
////////////////////////////////////  
private int getEncryptCoord(int coord)  
{  
    if(coord < 4)  
    {  
        return coord + 1;  
    }  
    else  
    {  
        return 0;  
    }  
}  
////////////////////////////////////
```

encryptText and encryptDi methods are connected methods, the goal of these methods is to encrypt the input text according to the Playfair cipher algorithm. The encryptText method will call encryptDi to encrypt each pair of the prepared input text matrix. In the encryptDi method the coordinates of the pair that were entered will be saved in coordsOfChar0 and coordsOfChar1, then we will initialize (X0,Y0) and (X1,Y1) and will be assigned with coordsOfChar0 and coordsOfChar1 values. Next is the encryption, if the pair were on different column and different row we are going to take the intersection of both, if the pair were in same column then we will apply a shift down with the use of a getEncryptcoord which is a helping method. Finally, the output will be appended to encryptedText String and we will be moving to the next pair until we finish.

## decryptText & decryptDi Methods:

```
////////////////////////////////////////
private String decryptText(String key, String inputText)//if the inputText
{
    char[][] matr = prepareInputText(inputText);
    Table playfairB = new Table(key);
    String decryptedText = "";

    for(int i = 0; i < matr.length; i++)
    {
        decryptedText = decryptedText + playfairB.decryptDi(matr[i]);
    }

    return decryptedText;
}
////////////////////////////////////////
```

```
////////////////////////////////////////
public String decryptDi(char[] matr)
{
    //matrix => nX2
    char[] output = new char[2];
    Integer[] coordsOfChar0 = idx.get(new Character(matr[0]));
    Integer[] coordsOfChar1 = idx.get(new Character(matr[1]));

    int X0 = coordsOfChar0[0].intValue();
    int X1 = coordsOfChar1[0].intValue();
    int Y0 = coordsOfChar0[1].intValue();
    int Y1 = coordsOfChar1[1].intValue();

    if(X0 != X1 && Y0 != Y1)//both different => taking the Intersection
    {
        output[0] = this.cipherMatrix[X0][Y1];
        output[1] = this.cipherMatrix[X1][Y0];
    }
    else if(X0 != X1 && Y0 == Y1)//same column => shift up
    {
        output[0] = this.cipherMatrix[getDecryptCoord(X0)][Y0];
        output[1] = this.cipherMatrix[getDecryptCoord(X1)][Y1];
    }
    else if(X0 == X1 && Y0 != Y1)//same row => shift left
    {
        output[0] = this.cipherMatrix[X0][getDecryptCoord(Y0)];
        output[1] = this.cipherMatrix[X1][getDecryptCoord(Y1)];
    }
    String strOutput = ArrayToString(output);
    return strOutput;
}
////////////////////////////////////////
```

## Helping Method:

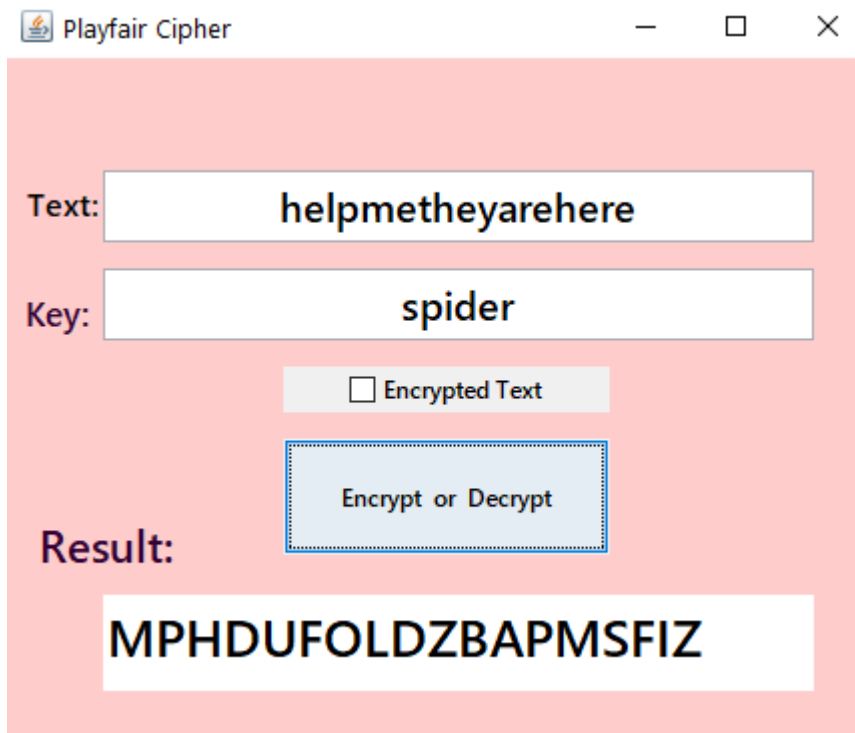
```
////////////////////////////////////  
private int getDecryptCoord(int coord)  
{  
    if(coord > 0)  
    {  
        return coord - 1;  
    }  
    else  
    {  
        return 4;  
    }  
}
```

decryptText and decryptDi methods are connected methods, the goal of these methods is to decrypt the input text according to the Playfair cipher algorithm. Same as encryptText and encryptDi, decryptText will call decryptDi to decrypt each pair of the input text matrix, the steps are similar to encryptText and encryptDi methods, however, in decryptDi the pair were in the same column then we will apply a shift up using the getDecryptCoord helping method or if the pair were in same row then we will apply a shift left using the getDecryptCoord helping method. The output will be converted and assigned to decryptedText String and we will be moving to the next pair until we finish.

## 5. Run Time Results (Screenshots)

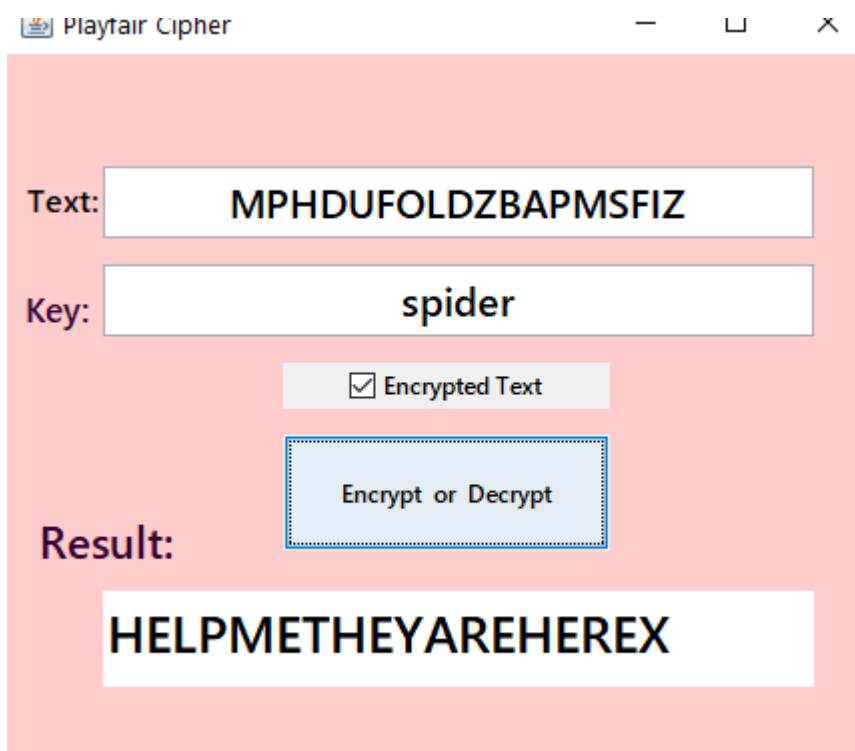
1)

Encrypt:



The screenshot shows a web application window titled "Playfair Cipher". It has a pink background. There are two input fields: "Text:" with the value "helpmetheyarehere" and "Key:" with the value "spider". Below these is a checkbox labeled "Encrypted Text" which is unchecked. A blue button labeled "Encrypt or Decrypt" is centered. Below the button, the "Result:" is displayed as "MPHDUFOLDZBAPMSFIZ".

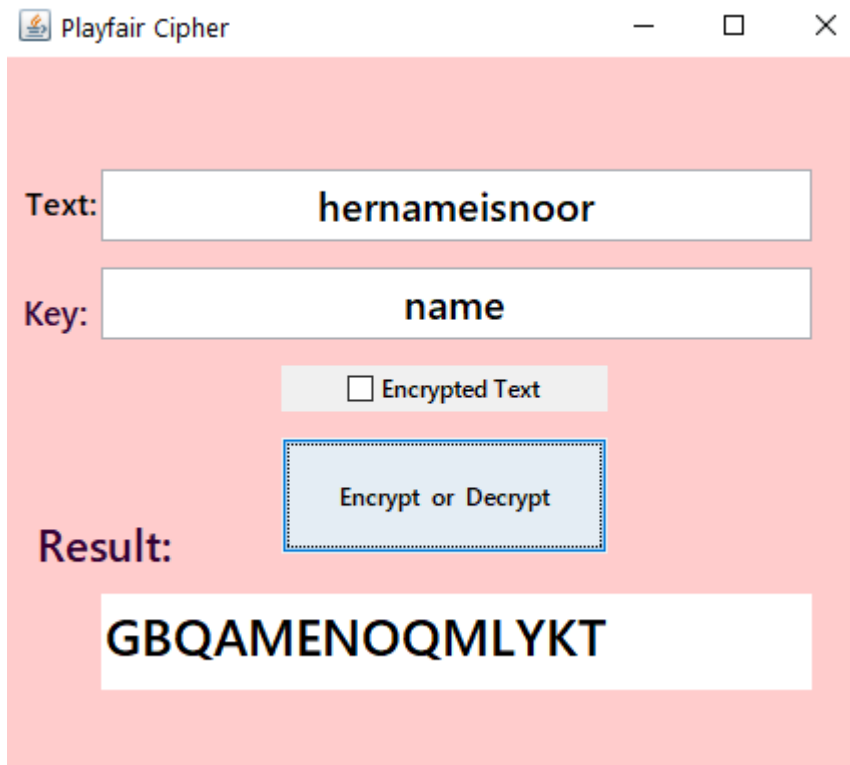
Decrypt:



The screenshot shows the same "Playfair Cipher" application window. The "Text:" field now contains "MPHDUFOLDZBAPMSFIZ" and the "Key:" field still contains "spider". The "Encrypted Text" checkbox is now checked. The "Result:" field displays "HELPMETHEYAREHEREX".

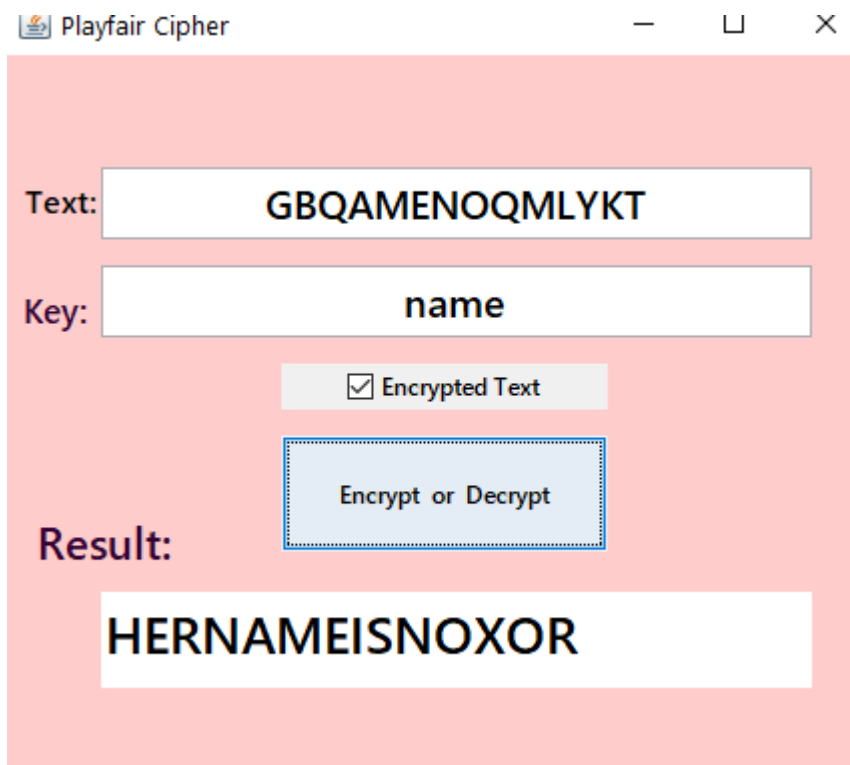
2)

Encrypt:



The screenshot shows a web application titled "Playfair Cipher". It has a light pink background. At the top, there is a header bar with the title and window control icons (minimize, maximize, close). Below the header, there are two input fields: "Text:" with the value "hernameisnoor" and "Key:" with the value "name". Below these fields, there is a checkbox labeled "Encrypted Text" which is currently unchecked. Below the checkbox is a button labeled "Encrypt or Decrypt". Below the button, there is a "Result:" label followed by a large white box containing the encrypted text "GBQAMENOQMLYKT".

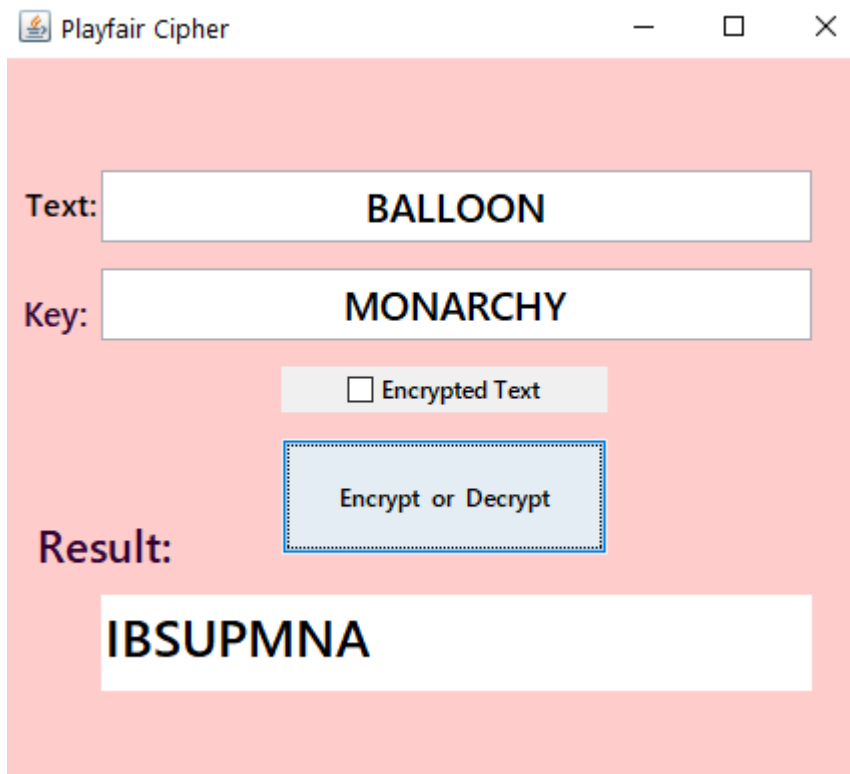
Decrypt:



The screenshot shows the same "Playfair Cipher" web application. In this state, the "Text:" field contains the encrypted text "GBQAMENOQMLYKT" and the "Key:" field still contains "name". The "Encrypted Text" checkbox is now checked. The "Encrypt or Decrypt" button remains the same. Below the button, the "Result:" label is followed by a large white box containing the decrypted text "HERNAMEISNOXOR".

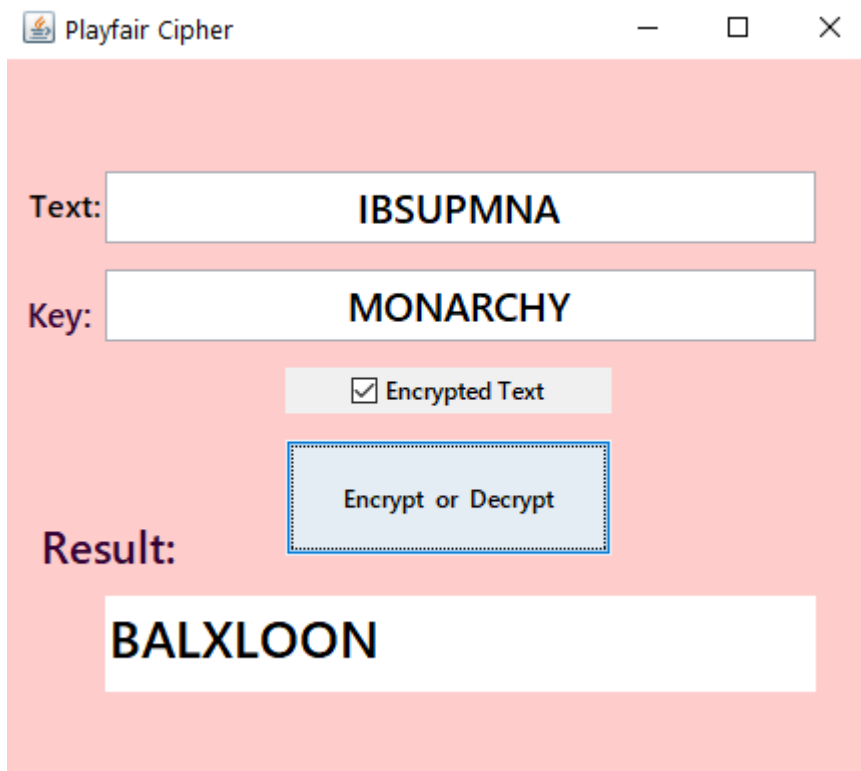
3)

Encrypt:



The screenshot shows a window titled "Playfair Cipher" with standard window controls. The interface has a light pink background. It contains two text input fields: "Text:" with the value "BALLOON" and "Key:" with the value "MONARCHY". Below these is a checkbox labeled "Encrypted Text" which is currently unchecked. A blue button with a dotted border is labeled "Encrypt or Decrypt". Below the button, the "Result:" label is followed by a text box containing the encrypted text "IBSUPMNA".

Decrypt:



The screenshot shows the same "Playfair Cipher" window. In this state, the "Text:" field contains "IBSUPMNA" and the "Key:" field contains "MONARCHY". The "Encrypted Text" checkbox is now checked. The "Encrypt or Decrypt" button remains the same. The "Result:" label is followed by a text box containing the decrypted text "BALXLOON".

## 6. References:

- 1- 2020. Cryptography Technique with Modular Multiplication Block Cipher and Playfair Cipher. 6th ed. [ebook] Indonesia: Universitas Pembangunan Panca Budi, p.range of pages. Available at: <[https://www.researchgate.net/profile/Robbi\\_Rahim3/publication/310021260\\_Cryptography\\_Technique\\_with\\_Modular\\_Multiplication\\_Block\\_Cipher\\_and\\_Playfair\\_Cipher/links/5827cdd308ae950ace6ce597/Cryptography-Technique-with-Modular-Multiplication-Block-Cipher-and-Playfair-Cipher.pdf](https://www.researchgate.net/profile/Robbi_Rahim3/publication/310021260_Cryptography_Technique_with_Modular_Multiplication_Block_Cipher_and_Playfair_Cipher/links/5827cdd308ae950ace6ce597/Cryptography-Technique-with-Modular-Multiplication-Block-Cipher-and-Playfair-Cipher.pdf)> [Accessed 18 November 2020].
- 2- Rahim, R. &. (2016). Cryptography technique with modular multiplication block cipher and playfair cipher. *Int. J. Sci. Res. Sci. Technol*, 2(6), 71-78.
- 3- Villafuerte, R. S. (2019). i3D-Playfair: An Improved 3D Playfair Cipher Algorithm. In *2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE)* (pp. 538-541). IEEE.
- 4- Amalia, B. M. (2018). File text security using hybrid cryptosystem with Playfair cipher algorithm and Knapsack Naccache-Stern algorithm. In *2nd International Conference on Computing and Applied Informatics 2017*. IOP Conf. Series. 978, p. 012114. *Journal of Physics: Conf.*