



*Scytale cipher*

# Computer Data Security & Privacy

***Programming Based Project***

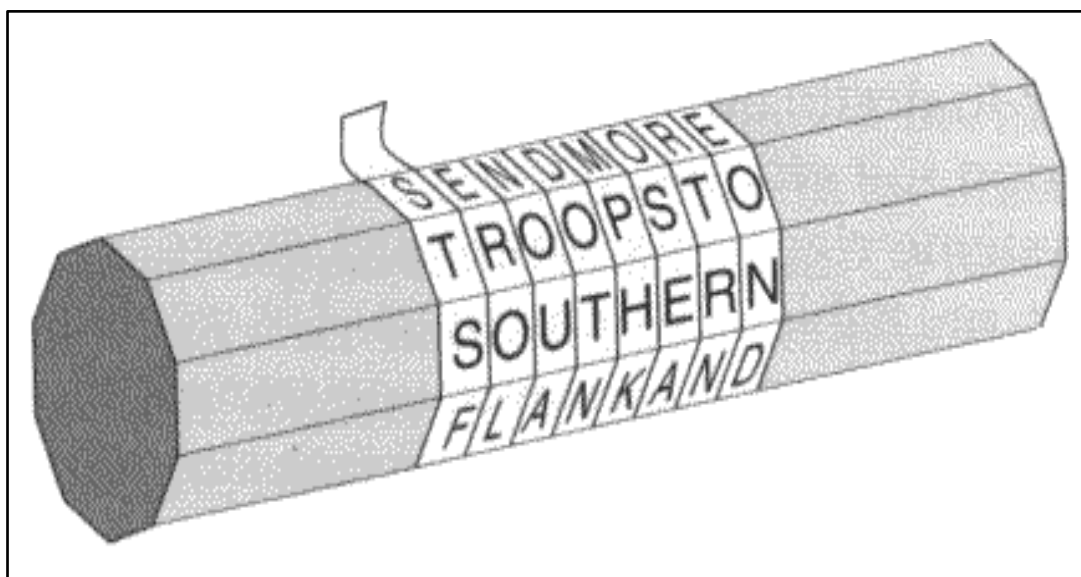


Instructor: Dr.Raouf Khan

Student Name	ID
Maryam Hussain	217035462
Salma Alomar	217019858

## 1. Introduction

Encryption is one of the techniques used to securely share the data between the sender and the receiver. By using encryption, the sender converts his/her message in a form that only the receiver can understand it and it seems ambiguous to any intruder that intercepts the message during its trip from source to destination. Encryption is categorized into two categories based on the key used in the algorithm to generate the encoded text which is Symmetric-key encryption and Asymmetric-key encryption. Symmetric-key encryption means that both the sender and receiver share a single privately key where they can use it for encryption and decryption while in Asymmetric-key encryption the sender and receiver use a different key for encryption and decryption the shared message. In this paper, we are implemented one of the symmetric encoding ciphers called Scytale cipher. The name refers to an old device used in the 4th century in ancient Greece to encrypt messages that send during the battle to officers and military. It is basically a strip of paper wrapped around a cylinder of wood see figure [1]. This paper shows jumble letters when it removed from the cylinder.in the destination, the receiver will wind the paper around the device that has the same diameter to read the message. In the next chapter, we will discuss in detail the implementing of this cipher using a java programming language (GUI).

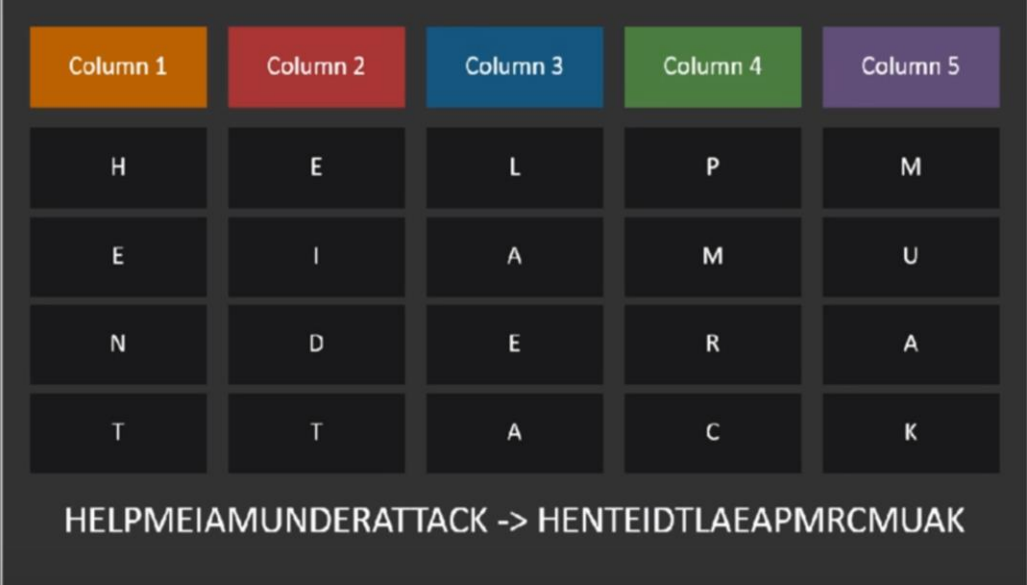


**Figure [1]: Scytale Cipher Device**

## 2. Description of the Cryptosystem Implemented

### 2.1.1 The Overall Idea to Encrypt the Plaintext

The idea considered as an improvement of the rail fence cipher (simple columnar transposition without using the key) by using a shared key that helps to hide the message content by changing the position of the letters. At the first, we will construct the tabular form (row and column). where the key represents the number of rows and the number of columns is equal to  $(\text{length of the message (without spaces)} / \text{number of rows (the key)})$ . Then, we start filling the table, row by row until the original message is over. Finally, obtaining the encoded text is done through lining up the letter from each column by moving left to right as shown in figure [2].



Column 1	Column 2	Column 3	Column 4	Column 5
H	E	L	P	M
E	I	A	M	U
N	D	E	R	A
T	T	A	C	K

HELPMEIAMUNDERATTACK -> HENTEIDTLAEAPMRCMUAK

**Figure [2]: demonstrate how lining up the letters to obtain the ciphertext.**

### 2.1.2 The Overall Idea to Decrypt the Ciphertext

The process is done at the receiver side in which the receiver will take advantage from knowing the key such as if the key equal to four, he will construct the table by writing the first four letters from top to bottom, then takes the next four letters and makes them adjacent to the first column and he will complete the same pattern until the ciphertext's letters finish. Now, the receiver can know what the message content by reading each row's letters.

## **2.2 Describe the Implementation through the Code**

To implement the encryption, the program prompts the users to enter their message and enter the number of rows (the key). We construct a method called "scytale\_Encode" with two-parameter to hold the message and the key-value for further manipulation. The algorithm is starting by compressed the plain text by removing the spaces between the letters and checking if the number of rows is less than or equal to zero or the number of rows is greater than or equal to the length of the message. If either case is meet the method will return the same plaintext. If not condition met the algorithm will be continued through checking the length of the message if it is dividable by two or not .we use "while loop" to check that condition and appending spaces as needed to adjust the length of the plaintext. Then, we store the number of columns in the "numOfCols" variable obtained by dividing the length of the plaintext by the numbers of rows to use it in the nested for loop which is considered the critical step to generate the ciphertext.

Similarly, the decryption is implemented by receiving the ciphertext and the number of rows from the user and passed as the "scytale\_Decode" method's arguments. The algorithm to get back the original text is starting by calculating the number of columns and store it in the "numOfCols" variable. Finally, we call the "scytale\_Encode" method to generate back the plaintext bypassing the ciphertext and "numOfCols" as the "scytale\_Encode" method's arguments and store the returned value in the "decodeText" variable.

## **3. Conclusion**

Many encryption/decryption techniques enable the secure message exchange between the source and receiver. We implemented one technique called Scytale cipher using a java programming language, which based on changing the position of the letters by using the key that both sender and receiver agreed to use to encrypt /decrypt the context of the message. Although

Scytale cipher is looking complicated to original people, it can be attacked by cryptanalysis by brute force technique.

#### 4. References

[1]<https://thestandardoutput.com/posts/a-scytale-algorithm-in-csharp/>

[2]The course's slides

#### 5. Appendix

Encryption\_Decryption\_System (main class)

```
package encryption_decryption_system;
```

```
import javax.swing.JFrame;
```

```
public class Encryption_Decryption_System {
```

```
    /**
```

```
     * @param args the command line arguments
```

```
    */
```

```
    public static void main(String[] args) {
```

```
        Main_Frame frm=new Main_Frame();
```

```
        frm.setVisible(true);
```

```
        frm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
        frm.setSize(1900,1100);
```

```
        frm.setLocationRelativeTo(null);
```

```
    }
```

```
}
```

## Main\_Frame class

```
package encryption_decryption_system;

import java.awt.Color;
import java.awt.Dimension;
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.Box;
import javax.swing.BoxLayout;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.border.EmptyBorder;

public class Main_Frame extends JFrame implements ActionListener{

    JLabel wellabel =new JLabel("Scytale Cipher");
    //JLabel services =new JLabel("Choose a Service:");
    JPanel welPanel=new JPanel();
    JPanel FristPanel=new JPanel();
    JPanel SecondPanel=new JPanel();
    JPanel BasePanel=new JPanel();

    JButton enc=new JButton("Encryption");
```

```

JButton dec=new JButton("Decryption");
Box box=Box.createVerticalBox();
public Main_Frame(){

// this.setLayout(new FlowLayout());
wellabel.setFont(new Font("DialogInput",1,70));
wellabel.setForeground(new Color(41,82,197));
//wellabel.setBorder(new EmptyBorder(50,200,50,200));
enc.setFont(new Font("DialogInput",Font.BOLD,60));
enc.add(Box.createRigidArea(new Dimension(400,100)));
enc.setBackground(new Color(37,115,31));
enc.setForeground(new Color(255,255,240));

dec.setFont(new Font("DialogInput",Font.BOLD,60));
dec.add(Box.createRigidArea(new Dimension(400,100)));
dec.setBackground(new Color(37,115,31));
dec.setForeground(new Color(255,255,240));

//services.setFont(new Font("DialogInput",1+2,50));
//services.setForeground(new Color(41,82,197));
BasePanel.setLayout(new BoxLayout(BasePanel,BoxLayout.Y_AXIS));
BasePanel.setBackground(new Color(180,229,204));
welPanel.setBackground(new Color(180,229,204));//255,192,219
BasePanel.add(Box.createVerticalStrut(50));
welPanel.add(wellabel);

BasePanel.add(Box.createVerticalStrut(30));
FristPanel.setLayout(new FlowLayout(FlowLayout.LEFT));
FristPanel.add(Box.createHorizontalStrut(100));
//FristPanel.add(services);
FristPanel.setBackground(new Color(180,229,204));
SecondPanel.setBorder(new EmptyBorder(50,200,50,200));
SecondPanel.setBackground(new Color(180,229,204));

```

```
SecondPanel.setLayout(new GridLayout(1,2,20,20) );
```

```
SecondPanel.add(enc);
```

```
SecondPanel.add(dec);
```

```
BasePanel.add(welPanel);
```

```
BasePanel.add(FristPanel);
```

```
BasePanel.add(SecondPanel);
```

```
BasePanel.add(Box.createVerticalStrut(30));
```

```
this.add(BasePanel);
```

```
enc.addActionListener(this);
```

```
dec.addActionListener(this);
```

```
}
```

```
@Override
```

```
public void actionPerformed(ActionEvent e) {
```

```
    if(e.getSource()==enc) {
```

```
        Encription_Cipher b=new Encription_Cipher();
```

```
        b.setVisible(true);
```

```
        b.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
```

```
        b.setSize(1800,1200);
```

```
        b.setLocationRelativeTo(null);
```

```
    }
```

```
    if(e.getSource()==dec) {
```

```
        Decryption_Cipher c=new Decryption_Cipher ();
```

```
        c.setVisible(true);
```

```
        c.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
```

```
        c.setSize(1800,1200);
```



```
c.setLocationRelativeTo(null);
```

```
}
```

```
}
```

```
}
```

## Encription\_Cipher class

```
package encryption_decryption_system;

import java.awt.Color;
import java.awt.Dimension;
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.Box;
import javax.swing.BoxLayout;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextArea;
import javax.swing.JTextField;

public class Encription_Cipher extends JFrame implements ActionListener{

    JLabel label =new JLabel("Encryption");
    JPanel topPanel=new JPanel();
    JPanel FristPanel=new JPanel();
    JPanel SecondPanel=new JPanel();
    JPanel thirdPanel=new JPanel();
    JPanel BasePanel=new JPanel();

    JLabel PlainText_lable =new JLabel("Enter The Plain Text:");
    static JTextArea PlainText_area=new JTextArea(15,80);
    JLabel key_lable =new JLabel("Enter The Key:");
    static JTextArea key_Text_area =new JTextArea(3,80);
```

```
JButton enc=new JButton("Encryption");
```

```
JButton Clear_b=new JButton("Clear");
```

```
public Encription_Cipher(){
```

```
    BasePanel.setLayout(new BoxLayout(BasePanel,BoxLayout.Y_AXIS));
```

```
    label.setFont(new Font("DialogInput",1,70));
```

```
    label.setForeground(new Color(41,82,197));
```

```
    //255,192,219
```

```
    BasePanel.add(Box.createVerticalStrut(50));
```

```
    topPanel.add(label);
```

```
    topPanel.setBackground(new Color(180,229,204));
```

```
    FristPanel.setBackground(new Color(180,229,204));
```

```
    BasePanel.setBackground(new Color(180,229,204));
```

```
    SecondPanel.setBackground(new Color(180,229,204));
```

```
    thirdPanel.setBackground(new Color(180,229,204));
```

```
    PlainText_lable.setFont(new Font("DialogInput",1,40));
```

```
    PlainText_lable.setForeground(new Color(41,82,197));
```

```
    PlainText_area.setWrapStyleWord(true);
```

```
    PlainText_area.setLineWrap(true);
```

```
    PlainText_area.setFont(new Font("DialogInput",1+2,20));
```

```
    key_Text_area.setFont(new Font("DialogInput",1+2,20));
```

```
    key_Text_area.setWrapStyleWord(true);
```

```
key_Text_area.setLineWrap(true);
```

```
FristPanel.setLayout(new FlowLayout(FlowLayout.LEFT));
```

```
FristPanel.add(Box.createHorizontalStrut(100));
```

```
FristPanel.add(PlainText_lable);
```

```
FristPanel.add(PlainText_area);
```

```
key_lable.setFont(new Font("DialogInput",1,40));
```

```
key_lable.setForeground(new Color(41,82,197));
```

```
SecondPanel.setLayout(new FlowLayout(FlowLayout.LEFT));
```

```
SecondPanel.add(Box.createHorizontalStrut(100));
```

```
SecondPanel.add(key_lable);
```

```
SecondPanel.add(Box.createHorizontalStrut(160));
```

```
SecondPanel.add(key_Text_area);
```

```
enc.setFont(new Font("DialogInput",Font.BOLD,40));
```

```
enc.add(Box.createRigidArea(new Dimension(400,150)));
```

```
enc.setBackground(new Color(37,115,31));
```

```
enc.setForeground(new Color(255,255,240));
```

```
Clear_b.setFont(new Font("DialogInput",Font.BOLD,40));
```

```
Clear_b.add(Box.createRigidArea(new Dimension(400,150)));
```

```
Clear_b.setBackground(new Color(37,115,31));
```

```
Clear_b.setForeground(new Color(255,255,240));
```

```
thirdPanel.add(enc);
```

```
thirdPanel.add(Box.createHorizontalStrut(50));
```

```
thirdPanel.add(Clear_b);
```

```
BasePanel.add(topPanel);
```

```
BasePanel.add(FristPanel);
BasePanel.add(SecondPanel);
BasePanel.add(thirdPanel);
this.add(BasePanel);

PlainText_area.setText("");
key_Text_area.setText("");

enc.addActionListener(this);
Clear_b.addActionListener(this);

}
```

```
@Override
public void actionPerformed(ActionEvent e) {
    //if clicks encryption
    //get plaintext and key
    String get_plaintext=PlainText_area.getText();
    String plaintext_key=key_Text_area.getText();

    if(e.getSource()==enc)
    {

        if(get_plaintext.isEmpty() || plaintext_key.isEmpty() ){
            JOptionPane.showMessageDialog(this,"Please, Fill both feilds");
            //System.out.print("empty");
        }
    }
}
```

```
}  
  
else{  
    Result_Encryption b=new Result_Encryption();  
    b.setVisible(true);  
    b.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);  
    b.setSize(1800,1200);  
    b.setLocationRelativeTo(null);  
}  
  
} //if  
  
if(e.getSource()==Clear_b){  
  
    key_Text_area.setText("");  
    PlainText_area.setText("");  
  
}  
  
} //actionPerformed
```

```
//class
```

## Result\_Encryption class

```
package encryption_decryption_system;

import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.Box;
import javax.swing.BoxLayout;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextArea;

public class Result_Encryption extends JFrame implements ActionListener {

    JLabel label =new JLabel("The Encrypted Text is");

    JTextArea encryptedText_area=new JTextArea(15,80);

    JPanel FristPanel=new JPanel();
    JPanel SecondPanel=new JPanel();
    JPanel thirdPanel=new JPanel();
    JPanel BasePanel=new JPanel();
    JButton Exit=new JButton("Exit");

    String get_plaintext=Encription_Cipher.PlainText_area.getText();
    int get_key=Integer.parseInt(Encription_Cipher.key_Text_area.getText());
    String encryption_result;
```

```
public Result_Encryption(){
    BasePanel.setLayout(new BoxLayout(BasePanel,BoxLayout.Y_AXIS));
    label.setFont(new Font("DialogInput",1,70));
    label.setForeground(new Color(41,82,197));

    BasePanel.add(Box.createVerticalStrut(50));
    FristPanel.add(label);

    FristPanel.setBackground(new Color(180,229,204));
    BasePanel.setBackground(new Color(180,229,204));
    SecondPanel.setBackground(new Color(180,229,204));
    thirdPanel.setBackground(new Color(180,229,204));

    eryptedText_area.setFont(new Font("DialogInput",1+2,20));
    eryptedText_area.setWrapStyleWord(true);
    eryptedText_area.setLineWrap(true);

    SecondPanel.add(eryptedText_area);

    Exit.setFont(new Font("DialogInput",Font.BOLD,40));
    Exit.add(Box.createRigidArea(new Dimension(400,150)));
    Exit.setBackground(new Color(37,115,31));
    Exit.setForeground(new Color(255,255,240));

    thirdPanel.add(Exit);
    BasePanel.add(FristPanel);
    BasePanel.add(SecondPanel);
    BasePanel.add(thirdPanel);

    this.add(BasePanel);
    Exit.addActionListener(this);
```



```
ecryption_result=scytale_Encode(get_plaintext,get_key);
encryptedText_area.setText(ecryption_result);
```

```
}
```

```
@Override
```

```
public void actionPerformed(ActionEvent e) {
```

```
    if(e.getSource()==Exit) {
```

```
        this.setVisible(false);
```

```
    } //if exist
```

```
    } //actionPerformed
```

```
//Encryption method
```

```
//to encode a plain text
```

```
public static String scytale_Encode(String plainText,int numOfRows ){
```

```
    String plainText_without_space="";
```

```
    String encodedText="";
```

```
    char[] chars=plainText.toCharArray();
```

```
    for (int i=0 ; i<chars.length;i++){
```

```
        if(chars[i]!=' '){
```

```
            plainText_without_space+=chars[i];
```

```
        }
```

```
    }
```

```
    System.out.println(plainText_without_space);
```

```

// check whether number of rows is greater than or= to plain text length

if(numOfRows>=plainText_without_space.length() || numOfRows<=0){
return plainText;}//if

// divide the plaintext length by number of rows
else{
    //if we have remainder, add space to the plain text
    while(plainText_without_space.length()%numOfRows!=0){
        plainText_without_space+="z";
    }//while

    //Calculate number of columns ()
    int numCols= plainText_without_space.length()/numOfRows;
    //int j=numCols;
    for(int i=0; i<numCols;i++){
        for(int j=i;j<plainText_without_space.length();j+=numCols){
            encodedText+=plainText_without_space.charAt(j);

        }//inner for

    }//outer for
}

return encodedText;

} //scytale_encode()
} //class

```

## Decryption\_Cipher class

```
package encryption_decryption_system;
```

```
import encryption_decryption_system.Encryption_Cipher.PlainText_area;    static
import java.awt.Color;
import java.awt.Dimension;
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.Box;
import javax.swing.BoxLayout;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextArea;
```

```
public class Decryption_Cipher extends JFrame implements ActionListener{
    JLabel label =new JLabel("Decryption");
    JPanel topPanel=new JPanel();
    JPanel FristPanel=new JPanel();
    JPanel SecondPanel=new JPanel();
```

```

JPanel thirdPanel=new JPanel();
JPanel BasePanel=new JPanel();

JLabel PlainText_lable =new JLabel("Enter The Cipher Text:");
static JTextArea PlainText_area=new JTextArea(15,80);
JLabel key_lable =new JLabel("Enter The Key:");
static JTextArea key_Text_area =new JTextArea(3,80);


JButton dec=new JButton("Decryption");

JButton Clear_b=new JButton("Clear");


public Decryption_Cipher(){
    BasePanel.setLayout(new BoxLayout(BasePanel,BoxLayout.Y_AXIS));
    label.setFont(new Font("DialogInput",1,70));
    label.setForeground(new Color(41,82,197));
    //255,192,219
    BasePanel.add(Box.createVerticalStrut(50));
    topPanel.add(label);


    topPanel.setBackground(new Color(180,229,204));
    FristPanel.setBackground(new Color(180,229,204));
    BasePanel.setBackground(new Color(180,229,204));
    SecondPanel.setBackground(new Color(180,229,204));
    thirdPanel.setBackground(new Color(180,229,204));


    PlainText_lable.setFont(new Font("DialogInput",1,40));
    PlainText_lable.setForeground(new Color(41,82,197));


    PlainText_area.setWrapStyleWord(true);
    PlainText_area.setLineWrap(true);

```

```
PlainText_area.setFont(new Font("DialogInput",1+2,20));
```

```
key_Text_area.setFont(new Font("DialogInput",1+2,20));
```

```
key_Text_area.setWrapStyleWord(true);
```

```
key_Text_area.setLineWrap(true);
```

```
FristPanel.setLayout(new FlowLayout(FlowLayout.LEFT));
```

```
FristPanel.add(Box.createHorizontalStrut(100));
```

```
FristPanel.add(PlainText_lable);
```

```
FristPanel.add(PlainText_area);
```

```
key_lable.setFont(new Font("DialogInput",1,40));
```

```
key_lable.setForeground(new Color(41,82,197));
```

```
SecondPanel.setLayout(new FlowLayout(FlowLayout.LEFT));
```

```
SecondPanel.add(Box.createHorizontalStrut(100));
```

```
SecondPanel.add(key_lable);
```

```
SecondPanel.add(Box.createHorizontalStrut(160));
```

```
SecondPanel.add(key_Text_area);
```

```
dec.setFont(new Font("DialogInput",Font.BOLD,40));
```

```
dec.add(Box.createRigidArea(new Dimension(400,150)));
```

```
dec.setBackground(new Color(37,115,31));
```

```
dec.setForeground(new Color(255,255,240));
```

```
Clear_b.setFont(new Font("DialogInput",Font.BOLD,40));
```

```
Clear_b.add(Box.createRigidArea(new Dimension(400,150)));
```

```
Clear_b.setBackground(new Color(37,115,31));
```

```
Clear_b.setForeground(new Color(255,255,240));
```

```
thirdPanel.add(dec);
thirdPanel.add(Box.createHorizontalStrut(50));
thirdPanel.add(Clear_b);
```

```
BasePanel.add(topPanel);
BasePanel.add(FristPanel);
BasePanel.add(SecondPanel);
BasePanel.add(thirdPanel);
this.add(BasePanel);
```

```
PlainText_area.setText("");
key_Text_area.setText("");
```

```
dec.addActionListener(this);
Clear_b.addActionListener(this);
```

```
}
```

```
@Override
```

```
public void actionPerformed(ActionEvent e) {
```

```
String get_cipherText=PlainText_area.getText();
String get_key=key_Text_area.getText();
```

```
if(e.getSource()==dec) {
```

```
if(get_cipherText.isEmpty() || get_key.isEmpty() ){
```

```
    JOptionPane.showMessageDialog(this,"Please, Fill both feilds");
```

```
    //System.out.print("empty");
```

```
}
```

```
else{
```

```
this.setVisible(false);
```

```
Result_Decryption b=new Result_Decryption();
```

```
b.setVisible(true);
```

```
b.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
```

```
b.setSize(1800,1200);
```

```
b.setLocationRelativeTo(null);} 
```

```
}
```

```
if(e.getSource()==Clear_b){
```

```
key_Text_area.setText("");
```

```
PlainText_area.setText("");
```

```
}
```

```
}
```

```
}
```

## Result\_Decryption class

```
package encryption_decryption_system;

import encryption_decryption_system.Result_Encryption.scytale_Encode;      static
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.Box;
import javax.swing.BoxLayout;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextArea;

public class Result_Decryption extends JFrame implements ActionListener{

    JLabel label =new JLabel("The Original Text is");

    JTextArea decryptedText_area=new JTextArea(15,80);

    JPanel FristPanel=new JPanel();

    JPanel SecondPanel=new JPanel();
```



```
JPanel thirdPanel=new JPanel();
JPanel BasePanel=new JPanel();
JButton Exit=new JButton("Exit");

String get_plaintext=Decryption_Cipher.PlainText_area.getText();
int get_key=Integer.parseInt(Decryption_Cipher.key_Text_area.getText());
String decryption_result;

public Result_Decryption(){
    BasePanel.setLayout(new BoxLayout(BasePanel,BoxLayout.Y_AXIS));
    label.setFont(new Font("DialogInput",1,70));
    label.setForeground(new Color(41,82,197));

    BasePanel.add(Box.createVerticalStrut(50));
    FristPanel.add(label);

    FristPanel.setBackground(new Color(180,229,204));
    BasePanel.setBackground(new Color(180,229,204));
    SecondPanel.setBackground(new Color(180,229,204));
    thirdPanel.setBackground(new Color(180,229,204));

    decryptedText_area.setFont(new Font("DialogInput",1+2,20));
    decryptedText_area.setWrapStyleWord(true);
    decryptedText_area.setLineWrap(true);

    SecondPanel.add(decryptedText_area);

    Exit.setFont(new Font("DialogInput",Font.BOLD,40));
    Exit.add(Box.createRigidArea(new Dimension(400,150)));
    Exit.setBackground(new Color(37,115,31));
    Exit.setForeground(new Color(255,255,240));
```

```

thirdPanel.add(Exit);
BasePanel.add(FristPanel);
BasePanel.add(SecondPanel);
BasePanel.add(thirdPanel);

this.add(BasePanel);
Exit.addActionListener(this);

decryption_result=scytale_Decode(get_plaintext,get_key);
decryptedText_area.setText(decryption_result);

}

@Override
public void actionPerformed(ActionEvent e) {
    if(e.getSource()==Exit) {

        this.setVisible(false);

    }
}

}

//to decode the encoded text
public static String scytale_Decode(String encodeText,int numOfRows ){

    String decodeText=" ";

    //Calculate number of coulms ()
    int numOfCols= encodeText.length()/numOfRows;

    //decode the encodeText

```

```
decodeText=Result_Encryption.scytale_Encode(encodeText,numOfCols).trim();
```

```
    return decodeText;
```

```
}//scytale_Decode()
```

```
}//class
```