Practice 2: CLASSICAL SYMMETRIC ENCRYPTION (Cont.)

2.1 OVERVIEW

2.1.1 Introduction

- Lab 2: Vigener Encryption Algorithm
- Practice time: class: 3 study hours, self-study: 3 study hours.
- Requirements: Students using Netbeans Software

2.1.2 Objective

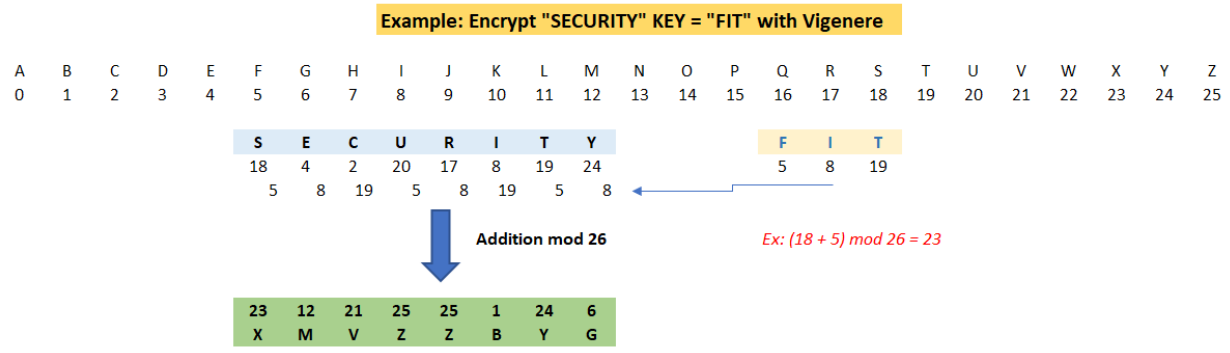
- This course provides students with knowledge of cryptographic algorithms and how they are used in today's world.
- The content emphasizes the principles, topics, approaches, and problem solving related to the underlying technologies and architectures of the field.

2.2 CONTENTS

2.2.1 Basic knowledge

The Vigenère cipher is also known as multiple encodings. This is a codification of this code that uses 26 more code tables. Do not beat that for a very long time. Although this code is gone, using the key with support is still advantageous.

Algorithm:



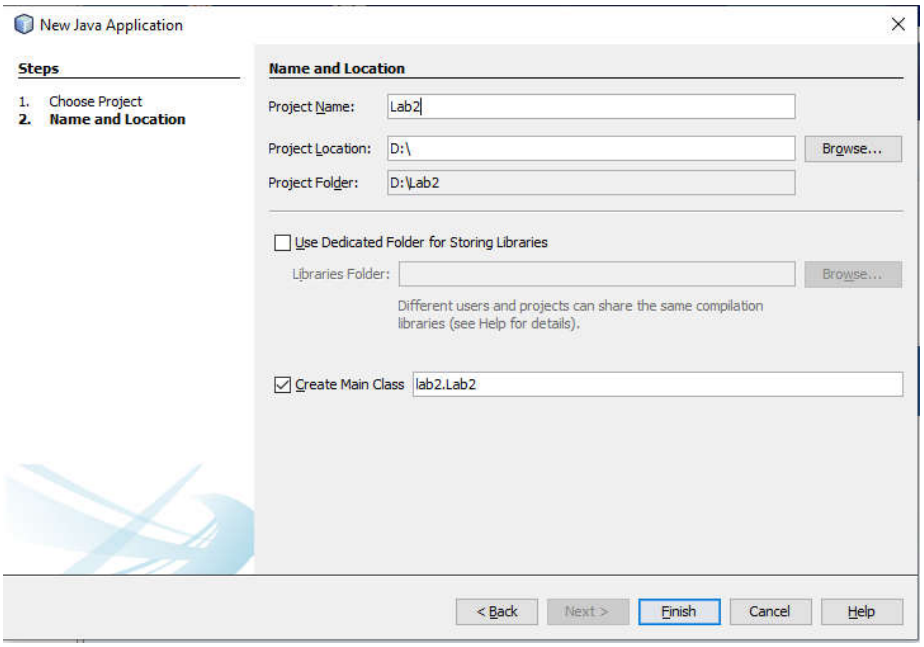
2.2.2 Vigenère Cipher

Write a program to encrypt and decrypt text with Vigenère encryption algorithm. The program can perform the following functions:

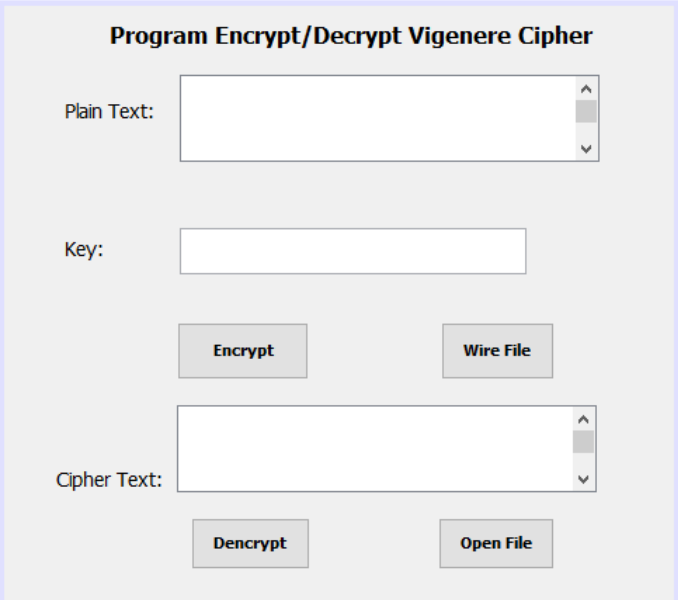
- Allow text input into the system.
- Allows entering text protection keys.
- Allows to write File and open File.

Guide:

❖ **Step 1: Create New Project -> Lab2**



❖ **Step 2: Design Form:**



❖ **Step 3: Write code for initialization function:**

```
 * @author Thang Nguyen
 */
public class Vigenere_Cipher extends javax.swing.JFrame {

        int Vig[][];
    public Vigenere_Cipher() {
        initComponents();
        Vig = new int [26][26];
        for (int i=0; i<26; i++)
            for(int j=0; j<26; j++)
                Vig [i][j] = (i+j)%26;
    }
```

❖ **Step 4: Write an event handler function:**

4.1 Encrypt function:

```
private String Encryption(String PlainText, String key)
{
    int n = PlainText.length();
    String CipherText = "";
    int k=0;
    for(int i=0; i<n; i++)
        if(Character.isLetter(PlainText.charAt(i)))
        {
            CipherText += Encrypt(PlainText.charAt(i), key.charAt(k));
            k++;
            k=k%key.length();
        }
        else
            CipherText+=PlainText.charAt(i);
    return CipherText;
}
```

```
char Encrypt (char x, char k)
{
    int xn = Character.toUpperCase(x) - 'A';
    int kn = Character.toUpperCase(k) - 'A';
    int yn = Vig[kn][xn];
    return (char) (yn + 'A');
}
```

4.2 Button Encrypt:

```
private void btnEncryptActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String PlainText = this.txtPlain.getText();
    String k = this.txtkey.getText();
    String CipherText = Encryption(PlainText, k);
    this.txtCipher.setText(CipherText);
}
```

4.3 Button WriteFile:

```
private void btnWiterFileActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try
{
    BufferedWriter bw = null;
    String fileName = "D:\\Lab2.txt";
    String s = txtPlain.getText();
    bw = new BufferedWriter(new FileWriter(fileName));
    bw.write(s);
    bw.close();
    JOptionPane.showMessageDialog(null, "Wrote File Success!!!!!");
}
catch (IOException ex)
{
    Logger.getLogger(Vigenere_Cipher.class.getName()).log(Level.SEVERE, null, ex);
}
}
```

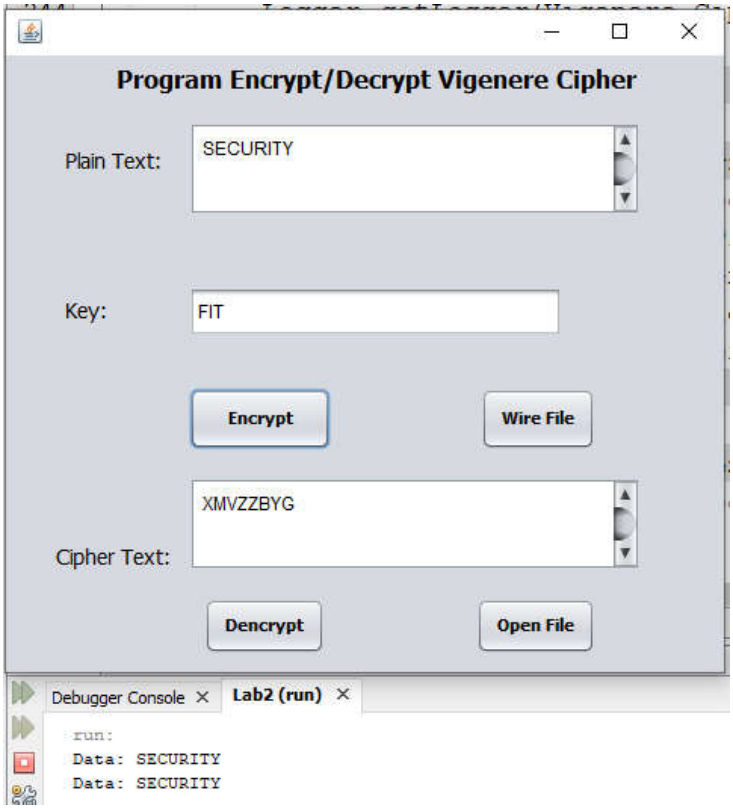
4.4 Button Decrypt:

```
private void btnDecryptActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
     String CipherText = this.txtPlain.getText();
    String k = this.txtkey.getText();
    String kt1 = "";
    int kn = k.length();
    for(int i=0; i<kn; i++)
    {
        kt1+=(char)(((26-(Character.toUpperCase(k.charAt(i))- 'A'))%26)+'A');
    }
    this.txtkey.setText(kt1);
    String PlainText = Encryption(CipherText, kt1);
    this.txtPlain.setText(PlainText);
}
```

4.6 Button OpenFile:

```
private void btnOpenFileActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
     try{
    BufferedReader br = null;
    String fileName = "D:\\Lab2.txt";
    br = new BufferedReader(new FileReader(fileName));
    StringBuffer sb = new StringBuffer();
    JOptionPane.showMessageDialog(null, "Opened File Success!!!!!!!!!!");
    char[] ca = new char[5];
    while(br.ready())
    {
        int len = br.read(ca);
        sb.append(ca, 0, len);
    }
    br.close();
    System.out.println("Data:" + " " + sb);
    String chuoi = sb.toString();
    txtPlain.setText(chuoi);
} catch (IOException ex) {
    Logger.getLogger(Vigenere_Cipher.class.getName()).log(Level.SEVERE, null, ex);
}
}
```

Result:



2.2.3 Rail Fence Cipher

Algorithm:

- Messages are written from left to right on the rails of an imaginary row diagonally from top to bottom.
- Diagonally from the bottom up when reaching the lowest column.
- And when you reach the top of the column, rewrite diagonally from top to bottom. Keep repeating like this until you write all the content of the message.

Example: *Plain text: Nice to meet you*

Key: 2

n c t m e y u

i e o e t o

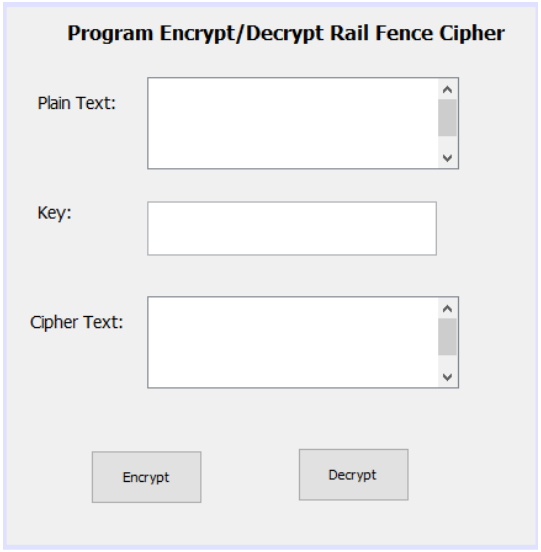
Cipher text: nctmeyuIeoeto

Write a program to encrypt and decrypt text with Vigenère encryption algorithm. The program can perform the following functions:

- Allow text input into the system.
- Allows entering text protection keys.
- Allows to write File and open File.

Guide:

❖ **Step 1:Create Form: Rail_Fence**



❖ **Step 2: write code for handle:**

➢ **2.1 Button Encrypt:**

```
private void btnEncryptActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    int k = Integer.valueOf(this.txtKey.getText());
    String s = this.txtPlain.getText();
    int n = s.length();
    int sd, sc;
    sd = k;
    sc = n/sd+1;
    char hr [][] = new char[sd][sc];
    int c,d;
    c=0;
    d=0;
    int sodu=n%sd;
    for(int i=0; i<n; i++)
    {
        hr[d][c]=s.charAt(i);
        d++;
        if(d==k)
        {
            c++;
            d=0;
        }
    }
    String kq="";
    int sokytu=sc;
    for(int i=0; i<sd; i++)
    {
        if(i>=sodu)
            sokytu=sc-1;
        for(int j=0; j<sokytu; j++)
            kq=kq+hr[i][j];
    }
    this.txtCipher.setText(kq);
}
```

➢ 2.2 Button Decrypt:

```
private void btnDecryptActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    int k = Integer.valueOf(this.txtKey.getText());
    String s = this.txtCipher.getText();
    int n = s.length();
    int sd,sc;
    sd=k;
    sc=n/sd+1;
    int sodu=n%sd;
    int sokytu=sc;
    int t=0;
    String kq="";
    char hr[][] = new char[sd][sc];
    for(int i=0; i<sd; i++)
    {
        if(i>=sodu)
            sokytu=sc-1;
        for(int j=0; j<sokytu; j++)
            hr[i][j] = s.charAt(t);
            t++;
    }
    int c,d;
    c=0;
    d=0;
    for(int i=0; i<n; i++)
    {
        kq+=hr[d][c];
        d++;
        if(d==k)
        {
            c++;
            d=0;
        }
    }
    this.txtPlain.setText(kq);
}
```

Result: