2.2 THUẬT TOÁN MÃ HÓA 3DES

Viết chương trình mã hóa và giải mã văn bán với thuật toán mã hóa 3DES.

Chương trình có thể thực hiện các chức năng sau:

Cho phép nhập văn bản vào hệ thống.

Cho phép nhập khóa báo vệ văn bán.

Cho phép ghi File và mở File.

2.2.1 Hướng dẫn thuật toán TRIPLEDES:

TripleDES một biến thể an toàn hơn của DES còn được gọi là DESede hay 3DES. TripleDES có tính bảo mật cao hơn DES do sử dụng 3 vòng DES với các khóa khác nhau. Vòng đầu tiên và vòng thứ ba là vòng mã hóa, vòng thứ hai là vòng giải mã. DESede có thể dùng hai hoặc ba khóa có độ dài 56, 112 hoặc 168. nếu dùng hai khóa thì khóa đầu tiên được dùng cho vòng thứ nhất và vòng thứ ba, khóa thứ hai dùng cho vòng thứ hai.

Mã hóa với ba khóa 56 bit (168 bit).

Bảng B1: 3DES Mã hóa với ba khóa 56 bit

NGƯỜI GỬI	NGƯỜI NHẬN
Bước 1: mã hóa plaintext bằng khóa nhất	Bước 1:giải mã bản mã với khóa thứ ba
Bước 2: mã hóa văn bản được tạo ra ở	Bước 2: giải mã văn bản được tạo ra ở
bước 1 bằng khóa thứ hai	bước 1 bằng khóa thứ hai
Bước 3: mã hóa văn bản được tạo ra ở	Bước 3: giải mã văn bản được tạo ra ở
bước 2 bằng khóa thứ ba, tạo ra bản	bước 2 bằng khóa thứ nhất, tạo ra
mã gửi cho người nhận.	bản gốc do người gửi gửi.

Mã hóa với hai khóa 56 bit (112 bit)

Bảng B2: 3DES Mã hóa với hai khóa 56 bit

NGƯỜI GỬI	NGƯỜI NHẬN
Bước 1: mã hóa plaintext bằng khóa	Bước 1:giải mã bản mã với khóa thứ
nhất	nhất
Bước 2: giải mã văn bản được tạo ra ở	Bước 2: mã hóa văn bản được tạo ra
bước 1 bằng khóa thứ hai	bước 1 bằng khóa thứ hai
Bước 3: mã hóa văn bản được tạo ra ở	Bước 3: giải mã văn bán được tạo ra
bước 2 bằng khóa thứ nhất, tạo ra bản mã gửi	bước 2 bằng khóa thứ nhất, tạo ra
cho người nhận.	bản gốc do người gửi gửi.

Mã hóa với một khóa 56 bit

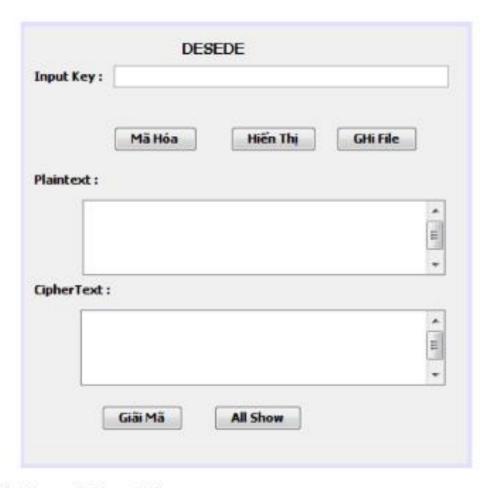
Bảng B3: 3DES Mã hóa với một khóa 56 bit

NGƯỜI GỬI	NGƯỜI NHẬN
Bước 1: mã hóa plaintext	Bước 1:giải mã bản mã nhận được từ người gửi.
Bước 2: giải mã văn bản được tạo ra ở bước 1	
Bước 3: mã hóa văn bản được tạo ra ở bước, tạo ra bản mã gửi cho người nhận.	

Mặc dù 3DES có tính bảo mật cao hơn DES, nhưng thực tế ít được sử dụng vì để tạo ra được bản mã phải chạy ba lần DES, nên tốc độ chậm, chiếm nhiều tài nguyên.

2.2.2 Hướng dẫn thực hành

Bước 1: Thiết Kế Form:



Bước 2: Viết hàm xử lý sự kiện

B2.1: Khai báo các biến sau

```
private static final String UNICODE_FORMAT = "UTF8";
public static final String DESEDE_ENCRYPTION_SCHEME = "DESede";
private KeySpec myKeySpec;
private SecretKeyFactory mySecretKeyFactory;
private Cipher cipher;
byte[] keyAsBytes;
private String myEncryptionKey;
private String myEncryptionScheme;
SecretKey key;
```

B2.2: Viết phương thức mã hóa encrypt

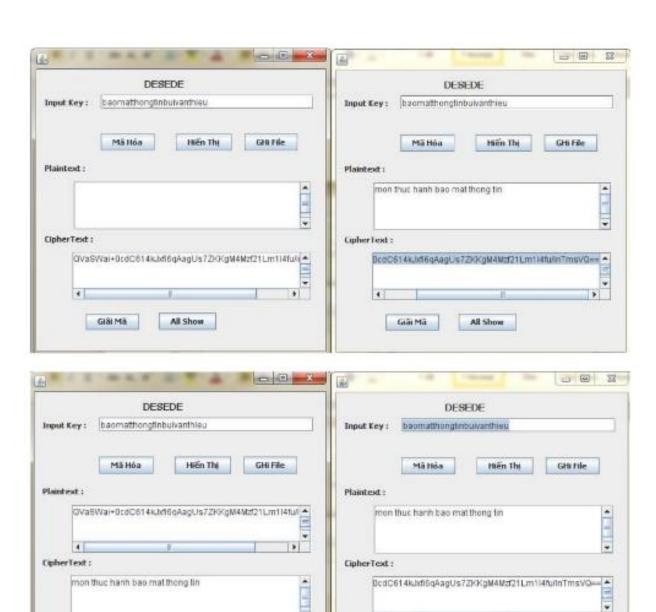
```
public String encrypt(String unencryptedString) (
   String encryptedString = null;
   try {
        cipher.init(Cipher.ENCRYFT_MODE, key);
        byte[] plainText = unencryptedString.getBytes(UNICODE_FORMAT);
        byte[] encryptedText = cipher.doFinal(plainText);
        BASE64Encoder base64encoder = new BASE64Encoder();
        encryptedString = base64encoder.encode(encryptedText);
    } catch (Exception e) {
        e.printStackTrace();
    }
    return encryptedString;
}
```

B2.3: Viết phương thức giãi mã decrypt

```
public String decrypt (String encryptedString) {
   String decryptedText=null;
   try {
      cipher.init(Cipher.DECRYPT_MODE, key);
      BASE64Decoder base64decoder = new BASE64Decoder();
      byte[] encryptedText = base64decoder.decodeBuffer(encryptedString);
      byte[] plainText = cipher.doFinal(encryptedText);
      String a= new String(plainText);
      System.out.println("chuoi plaintext :" + a);
      // decryptedText= bytes2String(plainText);
      decryptedText=a;
    } catch (Exception e) {
      e.printStackTrace();
    }
    return decryptedText;
}
```

B2.4 Viết hàm xứ lý sự kiện mã hóa

```
private void bntMaHoaActionPerformed(java.awt.event.ActionEvent evt) (
    try(
        // quot; Sanjaal.comsquot
    myEncryptionKey =txtkhoa.getText();
    myEncryptionScheme = DESEDE ENCRYPTION SCHEME;
    keyAsBytes = myEncryptionKey.getBytes(UNICODE_FORMAT);
    myKeySpec = new DESedeKeySpec(keyAsBytes);
    mySecretKeyFactory = SecretKeyFactory.getInstance(myEncryptionScheme);
    cipher = Cipher.getInstance(myEncryptionScheme);
    key = mySecretKeyFactory.generateSecret(myKeySpec);
    System.out.println(" khoa ma hoa k : " +" "+ key);
    // sử dụng lớp DESEDE EN
     String plainText=txtvanban.getText();
    // goi phương thức mã hoa
    String encrypted=encrypt(plainText);
    System.out.println("Encrypted Value :" + encrypted);
    txtmahoa.setText(encrypted);
    ) catch( Exception ex)()
```



GaiMa

All Show

Giài Mà

All Show