import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

import java.security.InvalidAlgorithmParameterException;

import java.security.InvalidKeyException;

import java.security.NoSuchAlgorithmException;

import java.security.spec.AlgorithmParameterSpec;

import javax.crypto.Cipher;

import javax.crypto.CipherInputStream;

import javax.crypto.CipherOutputStream;

import javax.crypto.KeyGenerator;

import javax.crypto.NoSuchPaddingException;

import javax.crypto.SecretKey;

import javax.crypto.spec.IvParameterSpec;

public class DES

{

//creating an instance of the Cipher class for encryption

private static Cipher encrypt;

//creating an instance of the Cipher class for decryption

private static Cipher decrypt;

//initializing vector

private static final byte[] initialization\_vector = { 22, 33, 11, 44, 55, 99, 66, 77 };

//main() method

public static void main(String[] args)

{

//path of the file that we want to encrypt

String textFile = "/home/20uad004/demo.txt";

//path of the encrypted file that we get as output

String encryptedData = "/home/20uad004/encrypteddata.txt";

//path of the decrypted file that we get as output

String decryptedData = "/home/20uad004/decrypteddata.txt";

try

{

//generating keys by using the KeyGenerator class

SecretKey scrtkey = KeyGenerator.getInstance("DES").generateKey();

AlgorithmParameterSpec aps = new IvParameterSpec(initialization\_vector);

//setting encryption mode

encrypt = Cipher.getInstance("DES/CBC/PKCS5Padding");

encrypt.init(Cipher.ENCRYPT\_MODE, scrtkey, aps);

//setting decryption mode

decrypt = Cipher.getInstance("DES/CBC/PKCS5Padding");

decrypt.init(Cipher.DECRYPT\_MODE, scrtkey, aps);

//calling encrypt() method to encrypt the file

encryption(new FileInputStream(textFile), new FileOutputStream(encryptedData));

//calling decrypt() method to decrypt the file

decryption(new FileInputStream(encryptedData), new FileOutputStream(decryptedData));

//prints the stetment if the program runs successfully

System.out.println("The encrypted and decrypted files have been created successfully.");

}

//catching multiple exceptions by using the | (or) operator in a single catch block

catch (NoSuchAlgorithmException | NoSuchPaddingException | InvalidKeyException | InvalidAlgorithmParameterException | IOException e)

{

//prints the message (if any) related to exceptions

e.printStackTrace();

}

}

//method for encryption

private static void encryption(InputStream input, OutputStream output)

throws IOException

{

output = new CipherOutputStream(output, encrypt);

//calling the writeBytes() method to write the encrypted bytes to the file

writeBytes(input, output);

}

//method for decryption

private static void decryption(InputStream input, OutputStream output)

throws IOException

{

input = new CipherInputStream(input, decrypt);

//calling the writeBytes() method to write the decrypted bytes to the file

writeBytes(input, output);

}

//method for writting bytes to the files

private static void writeBytes(InputStream input, OutputStream output)

throws IOException

{

byte[] writeBuffer = new byte[512];

int readBytes = 0;

while ((readBytes = input.read(writeBuffer)) >= 0)

{

output.write(writeBuffer, 0, readBytes);

}

//closing the output stream

output.close();

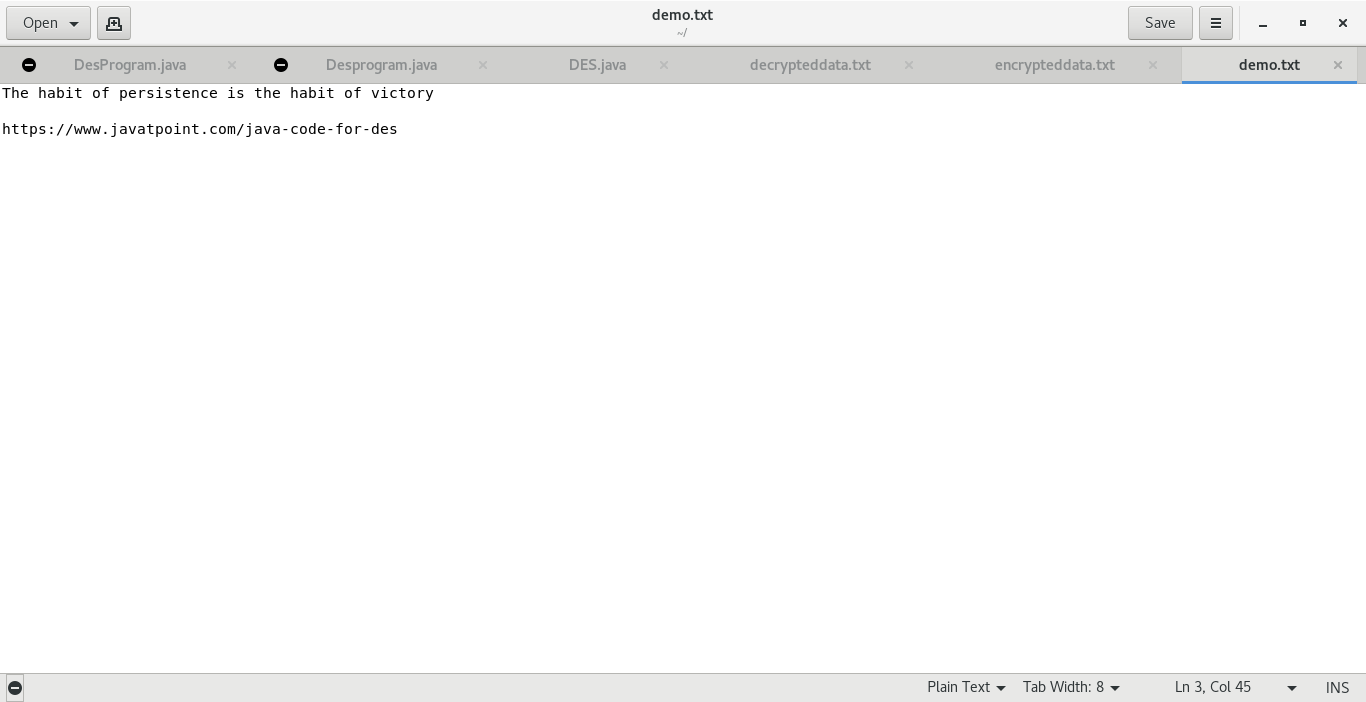
//closing the input stream

input.close();

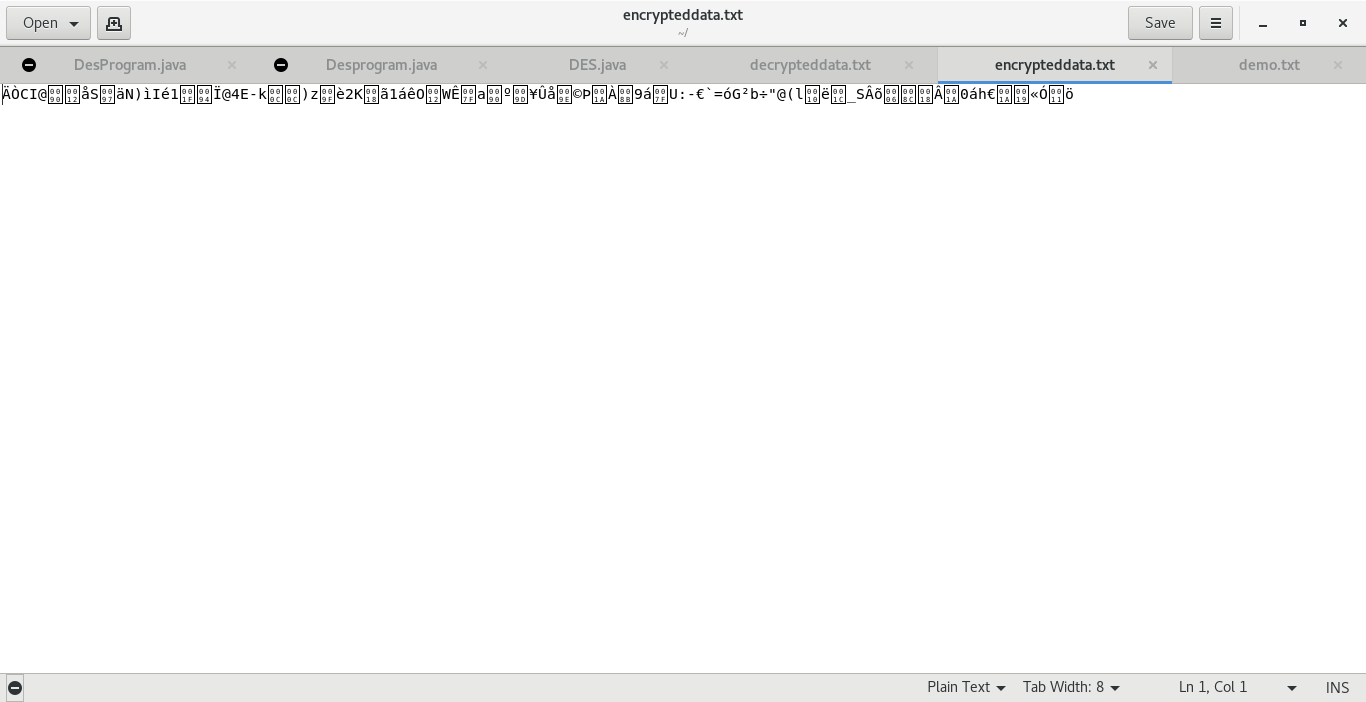
}

}

**Input File:demo.txt**

****

**Output File: encrypteddata.txt**

**Output File: decrypteddata.txt**

****