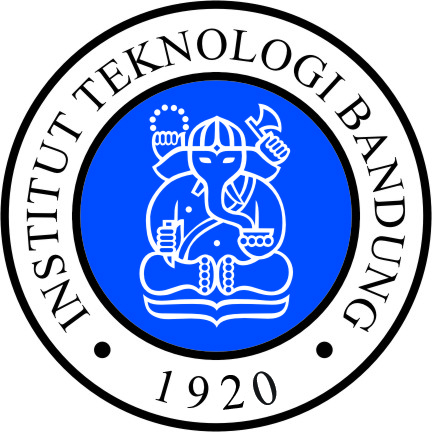
IF4020 Kriptografi

Tugas Kecil 1 Vigenere Cipher Applet



Disusun Oleh:

Timothy Pratama / 13512032

**Program Studi Teknik Informatika**

**Institut Teknologi Bandung**

**Jl. Ganesha 10, Bandung 40132**

# **Source Code Program Java**

\*source code dapat dilihat pada <https://github.com/timothy-pratama/VigenereCipherApplet>

## **Engine.java**

Engine.java adalah kelas yang mengimplementasikan *method* – *method* pada vigenere cipher.

package engine;

import java.io.File;

import java.io.FileNotFoundException;

import java.io.PrintWriter;

import java.io.UnsupportedEncodingException;

import java.util.Scanner;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\*

\* @author timothy.pratama

\*/

public class Engine {

//Attributes

private String ciphertext;

private String plaintext;

private String key; //maksimal 25 huruf

private int mode; //1: standard, 2: extended, 3: autokey

private int display; //1: apa adanya, 2: tanpa spasi, 3: dalam kelompok 5 huruf

private char[][] vigenereSquare26;

private char[][] vigenereSquare256;

//Methods

/\* Constructors \*/

public Engine() {

ciphertext = "";

plaintext = "";

key = "";

mode = 0;

display = 0;

createVigenereSquare();

}

public Engine(String ciphertext, String plaintext, String key, int mode, int display) {

this.ciphertext = ciphertext;

this.plaintext = plaintext;

this.key = key;

this.mode = mode;

this.display = display;

}

/\* Initialize Vigenere Square (26 and 256 characters) \*/

private void createVigenereSquare26() {

vigenereSquare26 = new char[26][26];

int c;

for (int b=0; b<26; b++) { //range 97 - 122

for (int k=0; k<26; k++) {

c = b + k + 97;

if (c > 122) {

c -= 26;

}

vigenereSquare26[b][k] = (char) c;

}

}

}

private void createVigenereSquare256() {

vigenereSquare256 = new char[256][256];

int c;

String output = "";

for (int b=0; b<256; b++) {

for (int k=0; k<256; k++) {

c = b + k;

if (c > 255) {

c -= 256;

}

vigenereSquare256[b][k] = (char) c;

}

}

}

private void createVigenereSquare() {

createVigenereSquare26();

createVigenereSquare256();

}

/\* Getter and Setter \*/

public String getCiphertext() {

String temp = "";

if(display == 1) {

temp = ciphertext;

} else if(display == 2) {

temp = ciphertext.replaceAll("\\s+","");

} else if(display == 3) {

temp = ciphertext.replaceAll("\\s+","");

int interval = 5;

int idx = 0;

String result = "";

while(idx + interval < temp.length()) {

result += temp.substring(idx, idx + interval) + " ";

idx += interval;

}

result += temp.substring(idx);

temp = result;

}

return temp;

}

public void setCiphertext(String ciphertext) {

this.ciphertext = ciphertext;

}

public String getPlaintext() {

return plaintext;

}

public void setPlaintext(String plaintext) {

this.plaintext = plaintext.toLowerCase();

}

public String getKey() {

return key;

}

public void setKey(String key) {

this.key = key;

}

public int getMode() {

return mode;

}

public void setMode(int mode) {

this.mode = mode;

}

public int getDisplay() {

return display;

}

public void setDisplay(int display) {

this.display = display;

}

public void readFile(String path) {

plaintext = "";

File file = new File(path);

try {

Scanner input = new Scanner(file);

do {

plaintext += input.nextLine() + "\n";

} while (input.hasNextLine());

plaintext = plaintext.toLowerCase();

} catch (FileNotFoundException ex) {

Logger.getLogger(Engine.class.getName()).log(Level.SEVERE, null, ex);

}

}

/\* save functions \*/

public void saveFile(String path) {

try {

PrintWriter writer = new PrintWriter(path, "UTF-8");

writer.print(ciphertext);

writer.close();

} catch (FileNotFoundException | UnsupportedEncodingException ex) {

Logger.getLogger(Engine.class.getName()).log(Level.SEVERE, null, ex);

}

}

/\* Cryptography functions \*/

public void encrypt() {

createKey();

ciphertext = "";

int plaintextLength = plaintext.length();

char c;

char k;

int j = 0;

if(mode == 1 || mode == 3) { //standard

for(int i = 0; i<plaintextLength; i++) {

c = plaintext.charAt(i);

if(c == ' ' || c == '\n') {

ciphertext += c;

} else {

k = key.charAt(j);

ciphertext += vigenereSquare26[((int) k) - 97][((int) c) - 97];

j++;

}

}

} else if (mode == 2 || mode == 4) { //extended

for(int i = 0; i<plaintextLength; i++) {

c = plaintext.charAt(i);

if(c == ' ' || c == '\n') {

ciphertext += c;

} else {

k = key.charAt(j);

ciphertext += vigenereSquare256[((int) k)][((int) c)];

j++;

}

}

}

}

public void decrypt() {

if(mode <= 2) {

createKey();

} else {

decodeKey();

}

plaintext = "";

int ciphertextLength = ciphertext.length();

char c;

char k;

int j = 0;

if(mode == 1 || mode == 3) { //standard

for(int i = 0; i<ciphertextLength; i++) {

c = ciphertext.charAt(i);

if(c == ' ' || c == '\n') {

plaintext += c;

} else {

k = key.charAt(j);

for(int l=0; l<26; l++) {

if(c == vigenereSquare26[((int)k)-97][l]) {

plaintext += (char) (l+97);

break;

}

}

j++;

}

}

} else if (mode == 2 || mode == 4) { //extended

for(int i = 0; i<ciphertextLength; i++) {

c = ciphertext.charAt(i);

if(c == ' ' || c == '\n') {

plaintext += c;

} else {

k = key.charAt(j);

for(int l=0; l<256; l++) {

if(c == vigenereSquare256[(int)k][l]) {

plaintext += (char) (l);

break;

}

}

j++;

}

}

}

}

private void createKey() {

int keyLength = key.length();

int plaintextLength = plaintext.replaceAll("\\s+","").length();

int ciphertextLength = ciphertext.replaceAll("\\s+","").length();

int length;

if(plaintextLength > ciphertextLength) {

length = plaintextLength;

} else {

length = ciphertextLength;

}

int j = 0;

if(mode <= 2) {

for(int i=key.length(); i<length; i++) {

key += key.charAt(j % keyLength);

j++;

}

}

else {

String temp = plaintext.replaceAll("\\s+","");

plaintextLength = temp.length();

for(int i=keyLength; i<plaintextLength; i++) {

while(plaintext.charAt(j) == ' ' || plaintext.charAt(j) == '\n') {

j++;

}

key += plaintext.charAt(j);

j++;

}

}

}

public void decodeKey() {

String tempCiphertext = ciphertext.replaceAll("\\s+","");

int ciphertextLength = tempCiphertext.length();

int initialKeyLength = key.length();

String currentKey = key;

String decrypt = "";

int counter = 0;

if(mode == 3) {//autokey - standard

while(currentKey.length() < ciphertextLength) {

decrypt = "";

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

char k = currentKey.charAt(i);

char c = tempCiphertext.charAt(i);

System.out.println("c= " + c);

for(int j=0; j<26; j++) {

if(c == vigenereSquare26[charToInteger(k)][j]) {

decrypt += (char) (j+97);

System.out.println("decrypt: " + decrypt);

break;

}

}

}

currentKey += decrypt.substring(counter);

counter+= initialKeyLength;

System.out.println("current key: " + currentKey);

if(currentKey.length() > ciphertextLength) {

currentKey = currentKey.substring(0, ciphertextLength);

}

}

key = currentKey;

} else { //autokey - extended

while(currentKey.length() < ciphertextLength) {

decrypt = "";

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

char k = currentKey.charAt(i);

char c = tempCiphertext.charAt(i);

System.out.println("c= " + c);

for(int j=0; j<256; j++) {

if(c == vigenereSquare256[(int)(k)][j]) {

decrypt += (char) (j);

System.out.println("decrypt: " + decrypt);

break;

}

}

}

currentKey += decrypt.substring(counter);

counter+= initialKeyLength;

System.out.println("current key: " + currentKey);

if(currentKey.length() > ciphertextLength) {

currentKey = currentKey.substring(0, ciphertextLength);

}

}

key = currentKey;

}

}

private int charToInteger(char c) {

int temp = ((int)c) - 97;

return temp;

}

public void EncryptFile(String path\_input, String path\_output, String key) {

File input = new File(path\_input);

FileInputStream in = null;

try {

in = new FileInputStream(input);

} catch (FileNotFoundException ex) {

Logger.getLogger(Engine.class.getName()).log(Level.SEVERE, null, ex);

}

byte[] isi = new byte[(int)input.length()];

byte[] output = new byte[(int)input.length()];

try {

int read = in.read(isi);

} catch (IOException ex) {

Logger.getLogger(Engine.class.getName()).log(Level.SEVERE, null, ex);

}

int panjang\_kunci = key.length();

int j=0;

for(int i=panjang\_kunci; i<input.length(); i++) {

key += key.charAt(j);

j++;

}

j=0;

for(byte b : isi) {

output[j] = (byte) ((b+key.charAt(j))%256);

j++;

}

FileOutputStream out = null;

try {

out = new FileOutputStream(new File(path\_output));

out.write(output);

out.close();

in.close();

} catch (FileNotFoundException ex) {

Logger.getLogger(Engine.class.getName()).log(Level.SEVERE, null, ex);

} catch (IOException ex) {

Logger.getLogger(Engine.class.getName()).log(Level.SEVERE, null, ex);

}

System.out.println("encrypt finished");

}

public void decryptFile(String path\_input, String path\_output, String key) {

File input = new File(path\_input);

FileInputStream in = null;

try {

in = new FileInputStream(input);

} catch (FileNotFoundException ex) {

Logger.getLogger(Engine.class.getName()).log(Level.SEVERE, null, ex);

}

byte[] isi = new byte[(int)input.length()];

byte[] output = new byte[(int)input.length()];

try {

int read = in.read(isi);

} catch (IOException ex) {

Logger.getLogger(Engine.class.getName()).log(Level.SEVERE, null, ex);

}

int panjang\_kunci = key.length();

int j=0;

for(int i=panjang\_kunci; i<input.length(); i++) {

key += key.charAt(j);

j++;

}

j=0;

for(byte b : isi) {

output[j] = (byte) ((b+256-key.charAt(j))%256);

j++;

}

FileOutputStream out = null;

try {

out = new FileOutputStream(new File(path\_output));

out.write(output);

out.close();

in.close();

} catch (FileNotFoundException ex) {

Logger.getLogger(Engine.class.getName()).log(Level.SEVERE, null, ex);

} catch (IOException ex) {

Logger.getLogger(Engine.class.getName()).log(Level.SEVERE, null, ex);

}

System.out.println("decrypt finished");

}

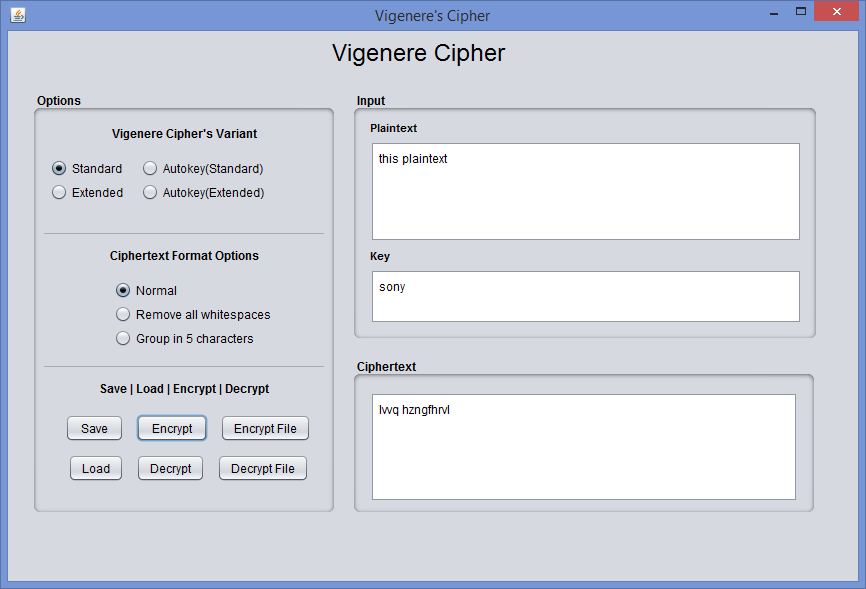
}

## **VigenereCipher.java**

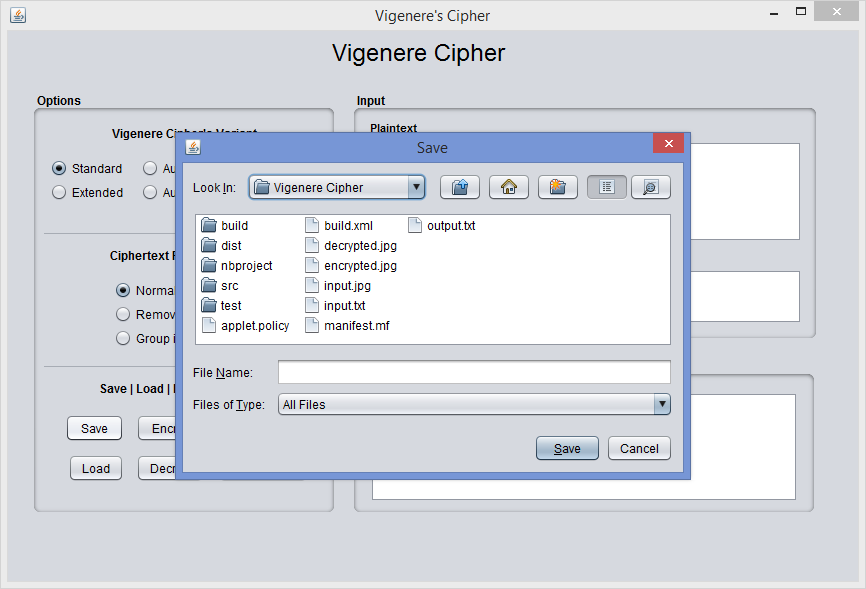
Kelas ini menggabungkan kelas Engine.java dengan GUI. Untuk *source code* GUI tidak ditulis di dalam laporan ini karena terlalu panjang (584 *lines*).

# Tampilan antarmuka program

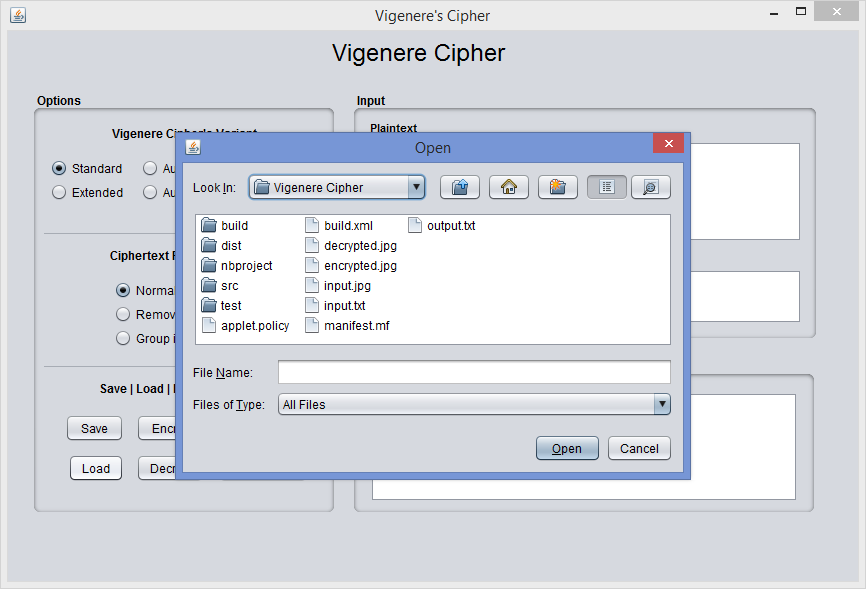
## Tampilan awal applet (beserta *encrypt* dan *decrypt*)



## Save *ciphertext* ke sebuah *file* eksternal



## Tampilan *load* *plaintext* dari *file* eksternal



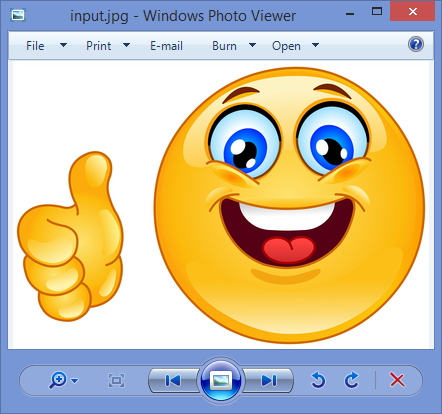
# Contoh *plaintext* dan *ciphertext*

Untuk contoh plaintext berukuran kecil, format ciphertext tidak menggunakan spasi, untuk yang berukuran sedang, format ciphertext dikelompokkan dalam 5 character, dan untuk ukuran yang besar, ciphertext tidak sesuai dengan format plaintext.

|  |  |  |  |
| --- | --- | --- | --- |
| **Variant**  **Vigenere**  **Cipher** | **Plaintext** | **Key** | **Ciphertext** |
| Standard | sekolah teknik elektro dan informatika | itb | axlwebpmfsgjsxmmduzheigjvypzfbbbli |
| We are currently opening opportunities for Indonesia undergraduate and master students to join our research projects in computer science | internship | ertvv pmyzt vgecf cwuqc obitf elbvx bvxww bjpvs waxwz nmult ztkeu hsamp vqfej gwyai cqxrk flvrd qahyi ewzmp zpati bblki avggf zhbbt zfvmv aul |
| Normally Mailman will remind you of your   mailing list passwords once every month although you can disable this if you prefer  This reminder will also include instructions on how to unsubscribe or change your account options  There is also a button on your options page that will email your current password to you | milis | zwcusxtj usutxif iqwt jqutvv kwf wx kwfz  emqwqfs ttal bidaoazoa gzkp mnqzj ugzbs idfpzcyt gzc umv oqkmjwm ltqd  qx kwf xjqnpz ltqd zwyqylwd ettd mtdw azkwcvq qyaldcnbaavd wf twh bg  gvdctekcqtq wc kzmvrm qacc iuowfvl axeqgza epwdm ta sxaz i tgbewf av  jwmd wabaavd xssm epsf ettd qulqd kwfz ugzcmff xlakiwcl la gzc |
| Extended | sekolah teknik elektro dan informatika | itb | ÜÙÍØàÃÑèÇÔâËÔÙÎÎßÖÛãÆÊâË×ÚÑÛáÃÝÝÍÊ |
| We are currently opening opportunities for Indonesia undergraduate and master students to join our research projects in computer science | internship | àÓÕ×× ÑèÚÛÕ ×âàÞá ÞØÖÒÞ ÐÝäÕá àçÝ×Ù Ý×ÙØØ ÝåÑ×Ô ØÜÙØÛ ÏèÖÍÕ ÛÕæÆÖ ãÔÜÎÑ ×ÒáÆå âØÚÜä ÞÒÙÓæ áç×Óß ÒÜãÚä àØÛÎÑ ÛÑÜÕä ÝÝÍÌä Ü×âÈá ÛãÝÝÕ Ûá×Î× ÜÖÍ |
| Normally Mailman will remind you of your   mailing list passwords once every month although you can disable this if you prefer  This reminder will also include instructions on how to unsubscribe or change your account options  There is also a button on your options page that will email your current password to you | milis | ÛØÞÖÔÙÕå ÖÔÖÕÙÊá äÒØÕ åÒÖÕ×× æØá ØÙ æØáÛ  àÎÒØÒáÔ ÕÕÜç ÝÊßÜêÜÛÐÜ âÛÌÑ ÎéÒÛå ÖâÛÝÔ ÊßáÑÛÞÚÕ âÛÞ ÖÎ× ÐÒæÎËØÎ çÕÒß  ÒÙ æØá ÙåÒÏÑÛ çÕÒß ÛØÚÒÚÍØß àÕÕß ÎÕßØ ÜÛÌØÞ×Ò ÒÚÜçßÞÏÝÜÜ×ß Øá ÕØã Ýâ  â×ßÞÕàÌÞÒÕÒ ØÞ ÌÛÎ×ÓÎ ìÜÞÞ ÊÖÐØá×ç ÜÙàÒâÛÜ àÑØßÎ ÕÜ ÔÙÜÛ Ê ÕâÝàØá Ü×  åØèß ØÜÝÜÜ×ß ÙÔÔÎ àÑÔá àÕÕß ÒÖÍÒß æØáÛ ÖâÛÞÎáá ÙÍÜæäØÞÍ çÜ âÛÞ |
| Standard, auto key | sekolah teknik elektro dan informatika | itb | axlgpkveergmurtooevywrblnswerokukt |
| We are currently opening opportunities for Indonesia undergraduate and master students to join our research projects in computer science | internship | ertvv pmyzt jxlps ryezr thanc gxhvv zwtht fkcal hvikn olvqh ftvsl uughe ejdav nekwt hgqnl lxfbh cqshk jxgno zpvji fnwgt jkurf ayyvx japks zry |
| Normally Mailman will remind you of your   mailing list passwords once every month although you can disable this if you prefer  This reminder will also include instructions on how to unsubscribe or change your account options  There is also a button on your options page that will email your current password to you | milis | zwcusyzp yatwkmn etxl eautyu cac bi wcof  rywfzzg ttag vlakpdrvk kbth wjrtc qjrkf mzgavura fco ihl rcuaoom lhjd  my fwm xwctyg klnw ixtqfuid evop rhaz tnndilr kymwvcplbfhu hv vbo hb  bbonpmpjccw qi kiebxg fohx eaqilnv qdnvhbh mpsew bz ecww s bflhoo ig  rchf bnhcfbh iiur lwaz absl xiitw cauz nsflvpn grwfpdrv lk mfx |
| Extended, auto key | Sekolah teknik elektro dan informatika | itb | ÜÙÍâÑÌ×àÆÓâÎÖÓÕÐÐà×ÚØÓÝÍÏÔØàÓÐæÖÌÕ |
| We are currently opening opportunities for Indonesia undergraduate and master students to join our research projects in computer science | internship | àÓÕ×× ÑèÚÛÕ åÙÍëÔ ÓÚàÛÓ ÕãÜéÞ âÙã×× ÛØÕãÕ áæÞÜÍ ã×ÊæÏ Ðç×ÒÉ áÕ×ÔÍ ÖÖâÉÆ àËßÂ× éÆæØÕ ãÈÒÏç çÙáÝã ÞÒÔãæ åÙâÏÐ ÛÑ×åä áÏØÈÕ åÌÖÓá ÜÚÚ×Ù åÜÑÌÔ ÛÓÚ |
| Normally Mailman will remind you of your   mailing list passwords once every month although you can disable this if you prefer  This reminder will also include instructions on how to unsubscribe or change your account options  There is also a button on your options page that will email your current password to you | milis | ÛØÞÖÔÚÛë ÚÂÕØæÎÏ àÕÙÍ àÜÖÕÚÖ ÞÜÞ ÝÊ òÞêá  ÓÚØáÛÛÈ ÕÕÜâ ×ÍÜæëßÓ×æ æÝÕÉ ØåÓÕÞ ÒåÓæá ÎÛâÜ×ÖÓÜ áÞê ÊÉç ÓÞÖÂÐÐÎ çÉËß  ÎÚ áØè ÙØÞÕÚâ æÍÏØ äÙÕÒáÖÊß à×ÐÑ ÓãÜÛ ÕÏÏßäÍÓ ÌÚèØ×ÞÑçÝáãÖ ã× ×Ýê ãÝ  ÝÝêéÑèÑåÞÄØ Òä ÌÊÆÝÙÈ áÐãÙ ÆÜÒäçÏ× Òßé×ãÝã èÑÔàØ ÝÛ ÆÞØØ Ô ÃáçãÐÐ äâ  íÞãá ÝéãÞáÝã äÊÖÓ çØÂÛ ÜÝÔÍ ÙäÊÕØ ÞÜÖÛ Ïîáç×Ñé âÓØáëßÓ× çæ èáÙ |

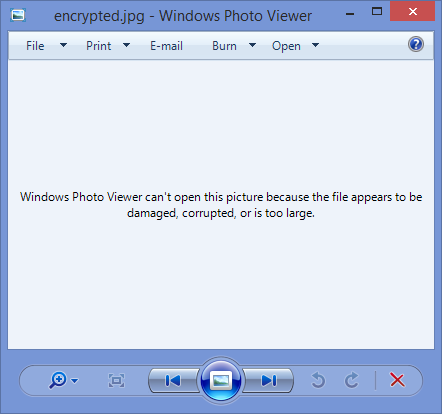
# Contoh *encrypt* dan *decrypt* sebuah file gambar (Bonus)

## **File input**



**File input.jpg**

## **File hasil enkripsi (dengan key = ‘kripto’)**



**File encrypted.jpg**

(tidak bisa dibuka karena *header file* menjadi terenkripsi)

## **File hasil dekrispi (dengan key = ‘kripto’)**



**File decrypted.jpg**