LAMPIRAN

Source Code Enkripsi dan Dekripsi SEAL

```
'-----------------------------------------------------------------------------------------------------------
'                                          file: Seal.frm
'                                   created by: Fredy Sadikun Halim
'-----------------------------------------------------------------------------------------------------------
Option Explicit

Dim i As Integer, j As Integer

Dim olefile As String
Dim encrypt As Boolean

'For the file actions
Dim path As String
Dim temp As String
Private lngInitializeValue As Long

Private Const StatusCaption As String = "SEAL Encryt Decrypt"
'************* Fungsi Encrypt ******************
Public Function EnCryption(plaintxt As String, pwd As String) As String
Dim temp As Integer, location As Integer, rccounter As Long, rccounter2 As Long, Aini As
Integer, bini As Integer
Dim a() As Byte, b() As Byte, c() As Byte, d() As Byte
Dim a2 As String, b2 As String, c2 As String, d2 As String
Dim l As Integer, m As Integer
Dim i As Integer, j As Integer
Dim TempArray1() As Byte, TempArray2() As Byte, TempArray3() As Byte, TempArray4() As
Byte, TempArray5() As Byte, TempArray6() As Byte, TempArray7() As Byte, TempArray8()
As Byte, TempArray9() As Byte, TempArray10() As Byte, TempArray11() As Byte,
TempArray12() As Byte, textenc As Byte, t510 As String, rotasi8 As String
Dim t() As String
Dim s() As String
Dim r() As String
Dim lngCRC32 As Long
```

```
 ReDim t(0 To 511) As String
 ReDim s(0 To 255) As String
 ReDim r(0 To 15) As String

Call sha_hash_msg(pwd, 1280)
For j = 0 To 509
t(j) = Trim(Str(bufA)) & Trim(Str(bufB)) & Trim(Str(bufC)) & Trim(Str(bufD)) & Trim(Str(bufE))
Next j

Call sha_hash_msg(pwd, 1279)
   t(510) = Trim(Str(bufA))
   t(511) = Trim(Str(bufB))

Call sha_hash_msg(pwd, 4095)
   s(0) = Trim(Str(bufB))
   s(1) = Trim(Str(bufC))
   s(2) = Trim(Str(bufD))
   s(3) = Trim(Str(bufE))

Call sha_hash_msg(pwd, 4096)
For m = 4 To 253
s(m) = Trim(Str(bufA)) & Trim(Str(bufB)) & Trim(Str(bufC)) & Trim(Str(bufD)) &
Trim(Str(bufE))
Next m

Call sha_hash_msg(pwd, 4094)
   s(254) = Trim(Str(bufA))
   s(255) = Trim(Str(bufB))

Call sha_hash_msg(pwd, 8190)
   r(0) = Trim(Str(bufB))
   r(1) = Trim(Str(bufC))
   r(2) = Trim(Str(bufD))
```

```
Call sha_hash_msg(pwd, 8192)
For l = 3 To 12
r(l) = Trim(Str(bufA)) & Trim(Str(bufB)) & Trim(Str(bufC)) & Trim(Str(bufD)) & Trim(Str(bufE))
Next l

Call sha_hash_msg(pwd, 8191)
  r(13) = Trim(Str(bufA))
  r(14) = Trim(Str(bufB))
  r(15) = Trim(Str(bufC))

Str2ByteArray plaintxt, TempArray1
Str2ByteArray plaintxt, TempArray2
Str2ByteArray plaintxt, TempArray3
Str2ByteArray plaintxt, TempArray4

For rccounter = 1 To UBound(TempArray1)
For l = 0 To 3
      location = (rccounter Mod Len(r(4 * l))) + 1
       'Plaintextbyte xor Keybyte

       TempArray1(rccounter) = TempArray1(rccounter) Xor Asc(Mid(r(4 * l), location, 1))
       TempArray5 = TempArray1
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray2)
For l = 0 To 3
      location = (rccounter Mod Len(r(4 * l + 1))) + 1
       'Plaintextbyte xor Keybyte

       TempArray2(rccounter) = TempArray2(rccounter) Xor Asc(Mid(r(4 * l + 1), location, 1))
       TempArray6 = TempArray2
Next l
Next rccounter
```

```
For rccounter = 1 To UBound(TempArray3)
For l = 0 To 3
      location = (rccounter Mod Len(r(4 * l + 2))) + 1
       'Plaintextbyte xor Keybyte

       TempArray3(rccounter) = TempArray3(rccounter) Xor Asc(Mid(r(4 * l + 2), location, 1))
       TempArray7 = TempArray3
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray4)
For l = 0 To 3
      location = (rccounter Mod Len(r(4 * l + 3))) + 1
       'Plaintextbyte xor Keybyte

       TempArray4(rccounter) = TempArray4(rccounter) Xor Asc(Mid(r(4 * l + 3), location, 1))
       TempArray8 = TempArray4
Next l
Next rccounter

'----------------------------

For rccounter = 1 To UBound(TempArray5)
For l = 0 To 127
      location = (rccounter Mod Len(t(4 * l))) + 1
       'Plaintextbyte xor Keybyte

       TempArray5(rccounter) = TempArray5(rccounter) Xor Asc(Mid(t(4 * l), location, 1))
       TempArray9 = TempArray5
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray6)
```

```
For l = 0 To 127
      location = (rccounter Mod Len(t(4 * l + 1))) + 1
       'Plaintextbyte xor Keybyte

       TempArray6(rccounter) = TempArray6(rccounter) Xor Asc(Mid(t(4 * l + 1), location, 1))
       TempArray10 = TempArray6
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray7)
For l = 0 To 127
      location = (rccounter Mod Len(t(4 * l + 2))) + 1
       'Plaintextbyte xor Keybyte

       TempArray7(rccounter) = TempArray7(rccounter) Xor Asc(Mid(t(4 * l + 2), location, 1))
       TempArray11 = TempArray7
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray8)
For l = 0 To 127
      location = (rccounter Mod Len(t(4 * l + 3))) + 1
       'Plaintextbyte xor Keybyte

       TempArray8(rccounter) = TempArray8(rccounter) Xor Asc(Mid(t(4 * l + 3), location, 1))
       TempArray12 = TempArray8
Next l
Next rccounter
'----------------------------
For rccounter = 1 To UBound(TempArray9)
For l = 0 To 63
      location = (rccounter Mod Len(s(4 * l))) + 1
       'Plaintextbyte xor Keybyte
```

```
        TempArray9(rccounter) = TempArray9(rccounter) Xor Asc(Mid(s(4 * l), location, 1))
        a = TempArray9
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray10)
For l = 0 To 63
       location = (rccounter Mod Len(s(4 * l + 1))) + 1
        'Plaintextbyte xor Keybyte

        TempArray10(rccounter) = TempArray10(rccounter) Xor Asc(Mid(s(4 * l + 1), location,
1))
        b = TempArray10
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray11)
For l = 0 To 63
       location = (rccounter Mod Len(s(4 * l + 2))) + 1
        'Plaintextbyte xor Keybyte

        TempArray11(rccounter) = TempArray11(rccounter) Xor Asc(Mid(s(4 * l + 2), location,
1))
        c = TempArray11
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray12)
For l = 0 To 63
       location = (rccounter Mod Len(s(4 * l + 3))) + 1
        'Plaintextbyte xor Keybyte

        TempArray12(rccounter) = TempArray12(rccounter) Xor Asc(Mid(s(4 * l + 3), location,
1))
```

```
        d = TempArray12
Next l
Next rccounter

a2 = StrConv(a, vbUnicode)
b2 = StrConv(b, vbUnicode)
c2 = StrConv(c, vbUnicode)
d2 = StrConv(d, vbUnicode)

EnCryption = a2 & b2 & c2 & d2

    lngInitializeValue = InitializeCRC32Table
    lngCRC32 = GenerateCRC32(EnCryption, lngInitializeValue)
    lblChecksum.Caption = "" & Hex$(lngCRC32)

Text2.Text = EnCryption
End Function
'************** Fungsi Decrypt ******************
Public Function decryption(plaintxt As String, pwd As String) As String
Dim temp As Integer, location As Integer, rccounter As Long, rccounter2 As Long, Aini As
Integer, bini As Integer
Dim a() As Byte, b() As Byte, c() As Byte, d() As Byte
Dim a2 As String, b2 As String, c2 As String, d2 As String
Dim l As Integer, m As Integer
Dim i As Integer, j As Integer
Dim lngLen As Long, lngLen2 As Long, output As String, output2 As String
Dim TempArray1() As Byte, TempArray2() As Byte, TempArray3() As Byte, TempArray4() As
Byte, TempArray5() As Byte, TempArray6() As Byte, TempArray7() As Byte, TempArray8()
As Byte, TempArray9() As Byte, TempArray10() As Byte, TempArray11() As Byte,
TempArray12() As Byte, textenc As Byte, t510 As String, rotasi8 As String
Dim t() As String
Dim s() As String
Dim r() As String
```

```
 ReDim t(0 To 511) As String
 ReDim s(0 To 255) As String
 ReDim r(0 To 15) As String

Call sha_hash_msg(pwd, 1280)
For j = 0 To 509
t(j) = Trim(Str(bufA)) & Trim(Str(bufB)) & Trim(Str(bufC)) & Trim(Str(bufD)) & Trim(Str(bufE))
Next j

Call sha_hash_msg(pwd, 1279)
   t(510) = Trim(Str(bufA))
   t(511) = Trim(Str(bufB))

Call sha_hash_msg(pwd, 4095)
   s(0) = Trim(Str(bufB))
   s(1) = Trim(Str(bufC))
   s(2) = Trim(Str(bufD))
   s(3) = Trim(Str(bufE))

Call sha_hash_msg(pwd, 4096)
For m = 4 To 253
s(m) = Trim(Str(bufA)) & Trim(Str(bufB)) & Trim(Str(bufC)) & Trim(Str(bufD)) &
Trim(Str(bufE))
Next m

Call sha_hash_msg(pwd, 4094)
   s(254) = Trim(Str(bufA))
   s(255) = Trim(Str(bufB))

Call sha_hash_msg(pwd, 8190)
   r(0) = Trim(Str(bufB))
   r(1) = Trim(Str(bufC))
   r(2) = Trim(Str(bufD))
```

```
Call sha_hash_msg(pwd, 8192)
For l = 3 To 12
r(l) = Trim(Str(bufA)) & Trim(Str(bufB)) & Trim(Str(bufC)) & Trim(Str(bufD)) & Trim(Str(bufE))
Next l

Call sha_hash_msg(pwd, 8191)
  r(13) = Trim(Str(bufA))
  r(14) = Trim(Str(bufB))
  r(15) = Trim(Str(bufC))

Str2ByteArray plaintxt, TempArray1
Str2ByteArray plaintxt, TempArray2
Str2ByteArray plaintxt, TempArray3
Str2ByteArray plaintxt, TempArray4

For rccounter = 1 To UBound(TempArray1)
For l = 0 To 63
      location = (rccounter Mod Len(s(4 * l))) + 1
       'Plaintextbyte xor Keybyte

       TempArray1(rccounter) = TempArray1(rccounter) Xor Asc(Mid(s(4 * l), location, 1))
       TempArray5 = TempArray1
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray2)
For l = 0 To 63
      location = (rccounter Mod Len(s(4 * l + 1))) + 1
       'Plaintextbyte xor Keybyte

       TempArray2(rccounter) = TempArray2(rccounter) Xor Asc(Mid(s(4 * l + 1), location, 1))
       TempArray6 = TempArray2
Next l
Next rccounter
```

```
For rccounter = 1 To UBound(TempArray3)
For l = 0 To 63
      location = (rccounter Mod Len(s(4 * l + 2))) + 1
       'Plaintextbyte xor Keybyte

       TempArray3(rccounter) = TempArray3(rccounter) Xor Asc(Mid(s(4 * l + 2), location, 1))
       TempArray7 = TempArray3
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray4)
For l = 0 To 63
      location = (rccounter Mod Len(s(4 * l + 3))) + 1
       'Plaintextbyte xor Keybyte

       TempArray4(rccounter) = TempArray4(rccounter) Xor Asc(Mid(s(4 * l + 3), location, 1))
       TempArray8 = TempArray4
Next l
Next rccounter
'-----------------------------
For rccounter = 1 To UBound(TempArray5)
For l = 0 To 127
      location = (rccounter Mod Len(t(4 * l))) + 1
       'Plaintextbyte xor Keybyte

       TempArray5(rccounter) = TempArray5(rccounter) Xor Asc(Mid(t(4 * l), location, 1))
       TempArray9 = TempArray5
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray6)
For l = 0 To 127
      location = (rccounter Mod Len(t(4 * l + 1))) + 1
```

```
        'Plaintextbyte xor Keybyte

        TempArray6(rccounter) = TempArray6(rccounter) Xor Asc(Mid(t(4 * l + 1), location, 1))
        TempArray10 = TempArray6
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray7)
For l = 0 To 127
      location = (rccounter Mod Len(t(4 * l + 2))) + 1
        'Plaintextbyte xor Keybyte

        TempArray7(rccounter) = TempArray7(rccounter) Xor Asc(Mid(t(4 * l + 2), location, 1))
        TempArray11 = TempArray7
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray8)
For l = 0 To 127
      location = (rccounter Mod Len(t(4 * l + 3))) + 1
        'Plaintextbyte xor Keybyte

        TempArray8(rccounter) = TempArray8(rccounter) Xor Asc(Mid(t(4 * l + 3), location, 1))
        TempArray12 = TempArray8
Next l
Next rccounter
'-----------------------------
For rccounter = 1 To UBound(TempArray9)
For l = 0 To 3
      location = (rccounter Mod Len(r(4 * l))) + 1
        'Plaintextbyte xor Keybyte

        TempArray9(rccounter) = TempArray9(rccounter) Xor Asc(Mid(r(4 * l), location, 1))
        a = TempArray9
```

```
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray10)
For l = 0 To 3
      location = (rccounter Mod Len(r(4 * l + 1))) + 1
       'Plaintextbyte xor Keybyte

       TempArray10(rccounter) = TempArray10(rccounter) Xor Asc(Mid(r(4 * l + 1), location,
1))
       b = TempArray10
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray11)
For l = 0 To 3
      location = (rccounter Mod Len(r(4 * l + 2))) + 1
       'Plaintextbyte xor Keybyte

       TempArray11(rccounter) = TempArray11(rccounter) Xor Asc(Mid(r(4 * l + 2), location,
1))
       c = TempArray11
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray12)
For l = 0 To 3
      location = (rccounter Mod Len(r(4 * l + 3))) + 1
       'Plaintextbyte xor Keybyte

       TempArray12(rccounter) = TempArray12(rccounter) Xor Asc(Mid(r(4 * l + 3), location,
1))
       d = TempArray12
Next l
```

```
Next rccounter

a2 = StrConv(a, vbUnicode)
b2 = StrConv(b, vbUnicode)
c2 = StrConv(c, vbUnicode)
d2 = StrConv(d, vbUnicode)

output = a2 & b2 & c2 & d2

For i = 1 To Len(output) / 16
decryption = decryption & Mid(output, i, 1)
Next i

Text2.Text = decryption
End Function

'************* Pengolahan File Encrypt-Decrypt ******************

Private Sub EnDeCryptfile()
Dim eNr As Integer
Dim DNr As Integer
Dim DNr2 As Integer
Dim crc As Integer
Dim Answer As Integer
Dim dCount As Double
Dim dRest As Double
Dim sTemp As String
Dim lngCRC32 As Long
Dim crcdec As String
Dim crccheck As String
Dim crchex As String
On Error GoTo Error
Dim Elapsed As Single
Dim EncodingRate As Single
```

```
i = 0: j = 0

If olefile = "" Then
path = SaveDialog(Me, "*.*", "Select file to encrypt", App.path)
Else
path = olefile
End If

If path = "" Then
   txtpwd.SetFocus
   Exit Sub
End If

'Status text
Form1.Caption = "Processing file..."

'Disable the Mousepointer
MousePointer = vbHourglass

'clear screen
Form1.Refresh

eNr = FreeFile
DNr = FreeFile + 1
DNr2 = FreeFile + 1
crc = FreeFile + 2
Open path For Binary As eNr

If LOF(eNr) = 0 Then
   Close eNr
   GoTo Error
End If
```

```
If encrypt Then

   'Peringatan jika terdapat file
   If FileExist(path + ".enc") Then

      Answer = MsgBox("File " & path + ".enc already exist - Continue?", vbYesNo, "Error")

      If Answer = vbYes Then
         SetAttr path + ".enc", vbArchive
      Else
         'Enable the Mousepointer
         MousePointer = vbDefault
         'Status
         Form1.Caption = StatusCaption
         'Set Focus
         txtpwd.SetFocus
         Exit Sub
      End If
   End If

   Open path + ".enc" For Binary As DNr
   Open path + ".crc" For Binary As crc



Else
   path = Left$(path, Len(path) - 4)

   'Peringatan jika terdapat file
   If FileExist(path) Then

      Answer = MsgBox("File " & path + " already exist - Continue?", vbYesNo, "Error")
```

```
        If Answer = vbYes Then
            SetAttr path, vbArchive
        Else
            'Enable the Mousepointer
            MousePointer = vbDefault
            'I'm ready!
            Form1.Caption = StatusCaption
            'Set Focus
            txtpwd.SetFocus
            Exit Sub
        End If
    End If

    Open path For Binary As DNr2
    Open path + ".crc" For Binary As crc
End If

 'Password -> SHA

    Call sha_hash_msg(txtpwd.Text, 0)
    sign_msg = Trim(Str(bufA)) & Trim(Str(bufB)) & Trim(Str(bufC)) & Trim(Str(bufD)) &
Trim(Str(bufE))

'Baca file dan encrypt/decrypt
If LOF(eNr) > 0 Then
    sTemp = Space(LOF(eNr))

    Get eNr, , sTemp

    If encrypt Then
    Elapsed = GetTickCount
    Put DNr, , EnCryption(sTemp, sign_msg)
    crcdec = lblChecksum.Caption
```

```
    Put crc, , crcdec
    EncodingRate = GetTickCount - Elapsed
    Label72.Caption = EncodingRate / 1000
    Else
    Elapsed = GetTickCount
        lngInitializeValue = InitializeCRC32Table
    lngCRC32 = GenerateCRC32(sTemp, lngInitializeValue)
    crchex = Hex$(lngCRC32)
    crccheck = Space(LOF(crc))
    Get crc, , crccheck
          If crchex = crccheck Then
       Put DNr2, , decryption(sTemp, sign_msg)
       lblChecksum.Caption = "OK!!"
       Else
       MsgBox "CRC file error", 0, "ERROR!"
       lblChecksum.Caption = "ERROR!!!"
       Text2.Text = ""
       End If
    EncodingRate = GetTickCount - Elapsed
    Label72.Caption = EncodingRate / 1000
    End If
    Text1.Text = sTemp
End If

sTemp = Empty

Close eNr
Close DNr
Close DNr2
Close crc

Close 1
Close 2
Close 3
```

```
'Enable the Mousepointer
MousePointer = vbDefault

'Status
Form1.Caption = StatusCaption
'Set Focus
txtpwd.SetFocus
Error:
End Sub

Private Sub Encryptb_Click()
olefile = ""
encrypt = True
Call check
End Sub

Private Sub Decrypt_Click()
   olefile = ""
   encrypt = False
   Call check
End Sub

Private Sub SEALrsa_Click()
form2.Show
Unload Me
End Sub

Private Sub TestSpeed_Click()
Dim GNr As Integer
Dim temp As String

'Time
Dim Elapsed As Single
Dim EncodingRate As Single
```

```
On Error GoTo Error

'Generate Temp Test Speed file
path = Environ("windir") & "\Temp\Test"
temp = String(300, "A")
GNr = FreeFile

Open path For Binary As GNr
Put GNr, , temp
Close GNr

'Encrypt Test Speed
olefile = path
encrypt = True
txtpwd = "11111"

Elapsed = GetTickCount
Call EnDeCryptfile

EncodingRate = GetTickCount - Elapsed

MsgBox "Encoding rate: " & 1000 / EncodingRate & " B/s, Encoding time : " & EncodingRate /
1000 & "s", vbInformation, "Speedtest"

If FileExist(path) Then Kill path
txtpwd = ""
Error:
End Sub


'This one is for Drag and Drop

Private Sub Encryptb_OLEDragDrop(Data As DataObject, Effect As Long, Button As Integer,
Shift As Integer, x As Single, y As Single)
```

```
Dim filecounter As Integer

For filecounter = 1 To Data.Files.Count
   If (GetAttr(Data.Files.Item(filecounter)) And vbDirectory) = 0 Then
      olefile = Data.Files(filecounter)
      encrypt = True
      Call check
      End If
Next
End Sub

Private Sub Decrypt_OLEDragDrop(Data As DataObject, Effect As Long, Button As Integer,
Shift As Integer, x As Single, y As Single)
Dim filecounter As Integer

For filecounter = 1 To Data.Files.Count
   If (GetAttr(Data.Files.Item(filecounter)) And vbDirectory) = 0 Then
         olefile = Data.Files(filecounter)
      encrypt = False
      Call check
      End If
Next
End Sub

Private Sub check()
'Error Check
If txtpwd.Text = "" Then
   MsgBox "Masukkan password terlebih dahulu sebelum encrypt atau decrypt file", 0,
"Error!"
   txtpwd.SetFocus
   Exit Sub
ElseIf Len(txtpwd) < 8 Then
End If
      Call EnDeCryptfile
```

```
End Sub

Private Sub Encryptm_Click()
olefile = ""
encrypt = True
Call check
End Sub

Private Sub Decryptm_Click()
olefile = ""
encrypt = False
Call check
End Sub

Private Sub Exitm_Click()
End
End Sub

Private Sub About_Click()
Load frmabout
Call ChangeStay(frmabout, 1)
frmabout.Show
End Sub

'Verify Password
Private Sub Ok_Click()
If temp = txtpwd Then
   Ok.Visible = False
   Cancel.Visible = False
   Call EnDeCryptfile
Else
   Ok.Visible = False
   Cancel.Visible = False
   Form1.Caption = StatusCaption
```

```
    txtpwd = ""
    txtpwd.SetFocus
    Exit Sub
End If

Form1.Caption = StatusCaption
End Sub

Private Sub Cancel_Click()
Ok.Visible = False
Cancel.Visible = False
Form1.Caption = StatusCaption
txtpwd = ""
txtpwd.SetFocus
End Sub

Private Sub TestSpeed2_Click()

'Time
Dim Elapsed2 As Single
Dim EncodingRate2 As Single

On Error GoTo Error
path = Environ("windir") & "\Temp\Test"
'Encrypt Test Speed
olefile = path + ".enc"
encrypt = False
txtpwd = "11111"

Elapsed2 = GetTickCount
Call EnDeCryptfile
EncodingRate2 = GetTickCount - Elapsed2
```

```
MsgBox "Decoding rate: " & 1000 / EncodingRate2 & " B/s, Decoding time : " &
EncodingRate2 / 1000 & "s", vbInformation, "Speedtest"

If FileExist(path) Then Kill path
If FileExist(path + ".enc") Then Kill path + ".enc"

txtpwd = ""
Error:
End Sub

'-----------------------------------------------------------------------------------------------------------
'                                          file: SealRSA.frm
'                                    created by: Fredy Sadikun Halim
'-----------------------------------------------------------------------------------------------------------
Option Explicit

Dim i As Integer, j As Integer

Dim olefile As String
Dim encrypt As Boolean

'For the file actions
Dim path As String
Dim temp As String

Private lngInitializeValue As Long

Private Const StatusCaption As String = "SEAL Encryt Decrypt"
'************* Fungsi Encrypt ******************
Public Function EnCryption(plaintxt As String, pwd As String) As String
Dim temp As Integer, location As Integer, rccounter As Long, rccounter2 As Long, Aini As
Integer, bini As Integer
Dim a() As Byte, b() As Byte, c() As Byte, d() As Byte
Dim a2 As String, b2 As String, c2 As String, d2 As String
```

```
Dim l As Integer, m As Integer
Dim i As Integer, j As Integer
Dim TempArray1() As Byte, TempArray2() As Byte, TempArray3() As Byte, TempArray4() As
Byte, TempArray5() As Byte, TempArray6() As Byte, TempArray7() As Byte, TempArray8()
As Byte, TempArray9() As Byte, TempArray10() As Byte, TempArray11() As Byte,
TempArray12() As Byte, textenc As Byte, t510 As String, rotasi8 As String
Dim t() As String
Dim s() As String
Dim r() As String
Dim lngCRC32 As Long

 ReDim t(0 To 511) As String
 ReDim s(0 To 255) As String
 ReDim r(0 To 15) As String


Call sha_hash_msg(pwd, 1280)
For j = 0 To 509
t(j) = Trim(Str(bufA)) & Trim(Str(bufB)) & Trim(Str(bufC)) & Trim(Str(bufD)) & Trim(Str(bufE))
Next j

Call sha_hash_msg(pwd, 1279)
  t(510) = Trim(Str(bufA))
  t(511) = Trim(Str(bufB))

Call sha_hash_msg(pwd, 4095)
  s(0) = Trim(Str(bufB))
  s(1) = Trim(Str(bufC))
  s(2) = Trim(Str(bufD))
  s(3) = Trim(Str(bufE))

Call sha_hash_msg(pwd, 4096)
For m = 4 To 253
```

```
s(m) = Trim(Str(bufA)) & Trim(Str(bufB)) & Trim(Str(bufC)) & Trim(Str(bufD)) &
Trim(Str(bufE))
Next m

Call sha_hash_msg(pwd, 4094)
  s(254) = Trim(Str(bufA))
  s(255) = Trim(Str(bufB))

Call sha_hash_msg(pwd, 8190)
  r(0) = Trim(Str(bufB))
  r(1) = Trim(Str(bufC))
  r(2) = Trim(Str(bufD))

Call sha_hash_msg(pwd, 8192)
For l = 3 To 12
r(l) = Trim(Str(bufA)) & Trim(Str(bufB)) & Trim(Str(bufC)) & Trim(Str(bufD)) & Trim(Str(bufE))
Next l

Call sha_hash_msg(pwd, 8191)
  r(13) = Trim(Str(bufA))
  r(14) = Trim(Str(bufB))
  r(15) = Trim(Str(bufC))

Str2ByteArray plaintxt, TempArray1
Str2ByteArray plaintxt, TempArray2
Str2ByteArray plaintxt, TempArray3
Str2ByteArray plaintxt, TempArray4

For rccounter = 1 To UBound(TempArray1)
For l = 0 To 3
      location = (rccounter Mod Len(r(4 * l))) + 1
       'Plaintextbyte xor Keybyte

       TempArray1(rccounter) = TempArray1(rccounter) Xor Asc(Mid(r(4 * l), location, 1))
```

```
        TempArray5 = TempArray1
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray2)
For l = 0 To 3
       location = (rccounter Mod Len(r(4 * l + 1))) + 1
        'Plaintextbyte xor Keybyte

        TempArray2(rccounter) = TempArray2(rccounter) Xor Asc(Mid(r(4 * l + 1), location, 1))
        TempArray6 = TempArray2
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray3)
For l = 0 To 3
       location = (rccounter Mod Len(r(4 * l + 2))) + 1
        'Plaintextbyte xor Keybyte

        TempArray3(rccounter) = TempArray3(rccounter) Xor Asc(Mid(r(4 * l + 2), location, 1))
        TempArray7 = TempArray3
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray4)
For l = 0 To 3
       location = (rccounter Mod Len(r(4 * l + 3))) + 1
        'Plaintextbyte xor Keybyte

        TempArray4(rccounter) = TempArray4(rccounter) Xor Asc(Mid(r(4 * l + 3), location, 1))
        TempArray8 = TempArray4
Next l
Next rccounter
```

```
'----------------------------

For rccounter = 1 To UBound(TempArray5)
For l = 0 To 127
      location = (rccounter Mod Len(t(4 * l))) + 1
       'Plaintextbyte xor Keybyte

       TempArray5(rccounter) = TempArray5(rccounter) Xor Asc(Mid(t(4 * l), location, 1))
       TempArray9 = TempArray5
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray6)
For l = 0 To 127
      location = (rccounter Mod Len(t(4 * l + 1))) + 1
       'Plaintextbyte xor Keybyte

       TempArray6(rccounter) = TempArray6(rccounter) Xor Asc(Mid(t(4 * l + 1), location, 1))
       TempArray10 = TempArray6
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray7)
For l = 0 To 127
      location = (rccounter Mod Len(t(4 * l + 2))) + 1
       'Plaintextbyte xor Keybyte

       TempArray7(rccounter) = TempArray7(rccounter) Xor Asc(Mid(t(4 * l + 2), location, 1))
       TempArray11 = TempArray7
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray8)
For l = 0 To 127
```

```
      location = (rccounter Mod Len(t(4 * l + 3))) + 1
       'Plaintextbyte xor Keybyte

       TempArray8(rccounter) = TempArray8(rccounter) Xor Asc(Mid(t(4 * l + 3), location, 1))
       TempArray12 = TempArray8
Next l
Next rccounter

'----------------------------

For rccounter = 1 To UBound(TempArray9)
For l = 0 To 63
      location = (rccounter Mod Len(s(4 * l))) + 1
       'Plaintextbyte xor Keybyte

       TempArray9(rccounter) = TempArray9(rccounter) Xor Asc(Mid(s(4 * l), location, 1))
       a = TempArray9
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray10)
For l = 0 To 63
      location = (rccounter Mod Len(s(4 * l + 1))) + 1
       'Plaintextbyte xor Keybyte

       TempArray10(rccounter) = TempArray10(rccounter) Xor Asc(Mid(s(4 * l + 1), location,
1))
       b = TempArray10
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray11)
For l = 0 To 63
      location = (rccounter Mod Len(s(4 * l + 2))) + 1
```

```
        'Plaintextbyte xor Keybyte

        TempArray11(rccounter) = TempArray11(rccounter) Xor Asc(Mid(s(4 * l + 2), location,
1))
        c = TempArray11
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray12)
For l = 0 To 63
       location = (rccounter Mod Len(s(4 * l + 3))) + 1
        'Plaintextbyte xor Keybyte

        TempArray12(rccounter) = TempArray12(rccounter) Xor Asc(Mid(s(4 * l + 3), location,
1))
        d = TempArray12
Next l
Next rccounter

a2 = StrConv(a, vbUnicode)
b2 = StrConv(b, vbUnicode)
c2 = StrConv(c, vbUnicode)
d2 = StrConv(d, vbUnicode)

EnCryption = a2 & b2 & c2 & d2

lngInitializeValue = InitializeCRC32Table
    lngCRC32 = GenerateCRC32(EnCryption, lngInitializeValue)
    lblChecksum.Caption = "" & Hex$(lngCRC32)

Text2.Text = EnCryption
End Function
'************* Fungsi Decrypt ******************
Public Function decryption(plaintxt As String, pwd As String) As String
```

```
Dim temp As Integer, location As Integer, rccounter As Long, rccounter2 As Long, Aini As Integer, bini As Integer
Dim a() As Byte, b() As Byte, c() As Byte, d() As Byte
Dim a2 As String, b2 As String, c2 As String, d2 As String
Dim l As Integer, m As Integer
Dim i As Integer, j As Integer
Dim lngLen As Long, lngLen2 As Long, output As String, output2 As String
Dim TempArray1() As Byte, TempArray2() As Byte, TempArray3() As Byte, TempArray4() As Byte, TempArray5() As Byte, TempArray6() As Byte, TempArray7() As Byte, TempArray8() As Byte, TempArray9() As Byte, TempArray10() As Byte, TempArray11() As Byte, TempArray12() As Byte, textenc As Byte, t510 As String, rotasi8 As String
Dim t() As String
Dim s() As String
Dim r() As String

 ReDim t(0 To 511) As String
 ReDim s(0 To 255) As String
 ReDim r(0 To 15) As String


Call sha_hash_msg(pwd, 1280)
For j = 0 To 509
t(j) = Trim(Str(bufA)) & Trim(Str(bufB)) & Trim(Str(bufC)) & Trim(Str(bufD)) & Trim(Str(bufE))
Next j

Call sha_hash_msg(pwd, 1279)
  t(510) = Trim(Str(bufA))
  t(511) = Trim(Str(bufB))

Call sha_hash_msg(pwd, 4095)
  s(0) = Trim(Str(bufB))
  s(1) = Trim(Str(bufC))
  s(2) = Trim(Str(bufD))
  s(3) = Trim(Str(bufE))
```

```
Call sha_hash_msg(pwd, 4096)
For m = 4 To 253
s(m) = Trim(Str(bufA)) & Trim(Str(bufB)) & Trim(Str(bufC)) & Trim(Str(bufD)) &
Trim(Str(bufE))
Next m

Call sha_hash_msg(pwd, 4094)
  s(254) = Trim(Str(bufA))
  s(255) = Trim(Str(bufB))

Call sha_hash_msg(pwd, 8190)
  r(0) = Trim(Str(bufB))
  r(1) = Trim(Str(bufC))
  r(2) = Trim(Str(bufD))

Call sha_hash_msg(pwd, 8192)
For l = 3 To 12
r(l) = Trim(Str(bufA)) & Trim(Str(bufB)) & Trim(Str(bufC)) & Trim(Str(bufD)) & Trim(Str(bufE))
Next l

Call sha_hash_msg(pwd, 8191)
  r(13) = Trim(Str(bufA))
  r(14) = Trim(Str(bufB))
  r(15) = Trim(Str(bufC))

Str2ByteArray plaintxt, TempArray1
Str2ByteArray plaintxt, TempArray2
Str2ByteArray plaintxt, TempArray3
Str2ByteArray plaintxt, TempArray4

For rccounter = 1 To UBound(TempArray1)
For l = 0 To 63
    location = (rccounter Mod Len(s(4 * l))) + 1
```

```
        'Plaintextbyte xor Keybyte

        TempArray1(rccounter) = TempArray1(rccounter) Xor Asc(Mid(s(4 * l), location, 1))
        TempArray5 = TempArray1
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray2)
For l = 0 To 63
      location = (rccounter Mod Len(s(4 * l + 1))) + 1
       'Plaintextbyte xor Keybyte

       TempArray2(rccounter) = TempArray2(rccounter) Xor Asc(Mid(s(4 * l + 1), location, 1))
       TempArray6 = TempArray2
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray3)
For l = 0 To 63
      location = (rccounter Mod Len(s(4 * l + 2))) + 1
       'Plaintextbyte xor Keybyte

       TempArray3(rccounter) = TempArray3(rccounter) Xor Asc(Mid(s(4 * l + 2), location, 1))
       TempArray7 = TempArray3
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray4)
For l = 0 To 63
      location = (rccounter Mod Len(s(4 * l + 3))) + 1
       'Plaintextbyte xor Keybyte

       TempArray4(rccounter) = TempArray4(rccounter) Xor Asc(Mid(s(4 * l + 3), location, 1))
       TempArray8 = TempArray4
```

```
Next l
Next rccounter

'----------------------------

For rccounter = 1 To UBound(TempArray5)
For l = 0 To 127
      location = (rccounter Mod Len(t(4 * l))) + 1
       'Plaintextbyte xor Keybyte

       TempArray5(rccounter) = TempArray5(rccounter) Xor Asc(Mid(t(4 * l), location, 1))
       TempArray9 = TempArray5
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray6)
For l = 0 To 127
      location = (rccounter Mod Len(t(4 * l + 1))) + 1
       'Plaintextbyte xor Keybyte

       TempArray6(rccounter) = TempArray6(rccounter) Xor Asc(Mid(t(4 * l + 1), location, 1))
       TempArray10 = TempArray6
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray7)
For l = 0 To 127
      location = (rccounter Mod Len(t(4 * l + 2))) + 1
       'Plaintextbyte xor Keybyte

       TempArray7(rccounter) = TempArray7(rccounter) Xor Asc(Mid(t(4 * l + 2), location, 1))
       TempArray11 = TempArray7
Next l
Next rccounter
```

```
For rccounter = 1 To UBound(TempArray8)
For l = 0 To 127
      location = (rccounter Mod Len(t(4 * l + 3))) + 1
       'Plaintextbyte xor Keybyte

       TempArray8(rccounter) = TempArray8(rccounter) Xor Asc(Mid(t(4 * l + 3), location, 1))
       TempArray12 = TempArray8
Next l
Next rccounter

'----------------------------

For rccounter = 1 To UBound(TempArray9)
For l = 0 To 3
      location = (rccounter Mod Len(r(4 * l))) + 1
       'Plaintextbyte xor Keybyte

       TempArray9(rccounter) = TempArray9(rccounter) Xor Asc(Mid(r(4 * l), location, 1))
       a = TempArray9
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray10)
For l = 0 To 3
      location = (rccounter Mod Len(r(4 * l + 1))) + 1
       'Plaintextbyte xor Keybyte

       TempArray10(rccounter) = TempArray10(rccounter) Xor Asc(Mid(r(4 * l + 1), location,
1))
       b = TempArray10
Next l
Next rccounter
```

```
For rccounter = 1 To UBound(TempArray11)
For l = 0 To 3
      location = (rccounter Mod Len(r(4 * l + 2))) + 1
       'Plaintextbyte xor Keybyte

       TempArray11(rccounter) = TempArray11(rccounter) Xor Asc(Mid(r(4 * l + 2), location,
1))
       c = TempArray11
Next l
Next rccounter

For rccounter = 1 To UBound(TempArray12)
For l = 0 To 3
      location = (rccounter Mod Len(r(4 * l + 3))) + 1
       'Plaintextbyte xor Keybyte

       TempArray12(rccounter) = TempArray12(rccounter) Xor Asc(Mid(r(4 * l + 3), location,
1))
       d = TempArray12
Next l
Next rccounter

a2 = StrConv(a, vbUnicode)
b2 = StrConv(b, vbUnicode)
c2 = StrConv(c, vbUnicode)
d2 = StrConv(d, vbUnicode)

output = a2 & b2 & c2 & d2

For i = 1 To Len(output) / 16
decryption = decryption & Mid(output, i, 1)
Next i

Text2.Text = decryption
```

```
End Function

'************* Pengolahan File Encrypt-Decrypt ******************

Private Sub EnDeCryptfile()
Dim eNr As Integer
Dim DNr As Integer
Dim DNr2 As Integer
Dim crc As Integer
Dim Answer As Integer
Dim dCount As Double
Dim dRest As Double
Dim sTemp As String
Dim encstr2 As String
Dim decstr2 As String
Dim lngCRC32 As Long
Dim crcdec As String
Dim crccheck As String
Dim crchex As String
Dim publickey As String
Dim crc2 As Integer
Dim Elapsed As Single
Dim EncodingRate As Single

On Error GoTo Error

i = 0: j = 0

If olefile = "" Then
path = SaveDialog(Me, "*.*", "Select file to encrypt", App.path)
Else
path = olefile
End If
```

```
If path = "" Then
   txtpwd.SetFocus
   Exit Sub
End If

'Status text
form2.Caption = "Processing file..."

'Disable the Mousepointer
MousePointer = vbHourglass

'clear screen
form2.Refresh

eNr = FreeFile
DNr = FreeFile + 1
DNr2 = FreeFile + 1
crc = FreeFile + 2
crc2 = FreeFile + 3
Open path For Binary As eNr

If LOF(eNr) = 0 Then
   Close eNr
   GoTo Error
End If

If encrypt Then


   'Peringatan jika terdapat file
   If FileExist(path + ".enc") Then

      Answer = MsgBox("File " & path + ".enc already exist - Continue?", vbYesNo, "Error")
```

```
        If Answer = vbYes Then
            SetAttr path + ".enc", vbArchive
        Else
            'Enable the Mousepointer
            MousePointer = vbDefault
            'Status
            form2.Caption = StatusCaption
            'Set Focus
            txtpwd.SetFocus
            Exit Sub
        End If
    End If

    Open path + ".enc" For Binary As DNr
    Open path + ".crc" For Binary As crc
    Open path + ".rsa" For Binary As crc2


Else
    path = Left$(path, Len(path) - 4)

    'Peringatan jika terdapat file
    If FileExist(path) Then

        Answer = MsgBox("File " & path + " already exist - Continue?", vbYesNo, "Error")

        If Answer = vbYes Then
            SetAttr path, vbArchive
        Else
            'Enable the Mousepointer
            MousePointer = vbDefault
            'I'm ready!
            form2.Caption = StatusCaption
            'Set Focus
```

```
            txtpwd.SetFocus
            Exit Sub
        End If
    End If

    Open path For Binary As DNr2
    Open path + ".crc" For Binary As crc
    Open path + ".rsa" For Binary As crc2
End If

 'Password -> SHA

    Call sha_hash_msg(publickey, 0)
    sign_msg = Trim(Str(bufA)) & Trim(Str(bufB)) & Trim(Str(bufC)) & Trim(Str(bufD)) &
Trim(Str(bufE))

'Baca file dan encrypt/decrypt
If LOF(eNr) > 0 Then
    sTemp = Space(LOF(eNr))
'RSA algoritma : p=43 q=67 n=2881
    Get eNr, , sTemp
    If encrypt Then
        Elapsed = GetTickCount
        encstr2 = "password"
        sign_msg = encstr2
        publickey = enc(encstr2, txtpwd2, 2881)
        Put DNr, , EnCryption(sTemp, sign_msg)
        crcdec = lblChecksum.Caption
        Put crc, , crcdec
        Put crc2, , publickey
        EncodingRate = GetTickCount - Elapsed
        Label72.Caption = EncodingRate / 1000
    Else
        Elapsed = GetTickCount
```

```
        lngInitializeValue = InitializeCRC32Table
        lngCRC32 = GenerateCRC32(sTemp, lngInitializeValue)
        crchex = Hex$(lngCRC32)
        crccheck = Space(LOF(crc))
        Get crc, , crccheck
        publickey = Space(LOF(crc2))
        Get crc2, , publickey
        decstr2 = dec(publickey, txtpwd2, 2881)
        sign_msg = decstr2
        If crchex = crccheck Then
           Put DNr2, , decryption(sTemp, sign_msg)
           lblChecksum.Caption = "OK!!"
        Else
           MsgBox "CRC file error", 0, "ERROR!"
           lblChecksum.Caption = "ERROR!!!"
           Text2.Text = ""
        End If
        EncodingRate = GetTickCount - Elapsed
        Label72.Caption = EncodingRate / 1000
    End If
    Text1.Text = sTemp


End If

sTemp = Empty

Close eNr
Close DNr
Close DNr2
Close crc
Close crc2
```

```
Close 1
Close 2
Close 3
Close 4

'Enable the Mousepointer
MousePointer = vbDefault

'Status
form2.Caption = StatusCaption

'Set Focus
txtpwd.SetFocus

Error:
End Sub

Private Sub Encryptb_Click()
olefile = ""
encrypt = True

Call check
End Sub

Private Sub Decrypt_Click()
   olefile = ""
   encrypt = False

   Call check
End Sub

Private Sub key_Click()
frmStepTwo.Show
End Sub
```

```
Private Sub Seal_Click()
Form1.Show
Unload Me
End Sub

Private Sub TestSpeed_Click()
Dim GNr As Integer
Dim temp As String

'Time
Dim Elapsed As Single
Dim EncodingRate As Single

On Error GoTo Error

'Generate Temp Test Speed file
path = Environ("windir") & "\Temp\Test"

temp = String(300, "A")

GNr = FreeFile

Open path For Binary As GNr
Put GNr, , temp
Close GNr


'Encrypt Test Speed
olefile = path
encrypt = True
txtpwd = "11111"

Elapsed = GetTickCount
```

```
Call EnDeCryptfile

EncodingRate = GetTickCount - Elapsed

MsgBox "On your System you get: " & 1000 / EncodingRate & " B/s", vbInformation,
"Speedtest"

If FileExist(path) Then Kill path
If FileExist(path + ".enc") Then Kill path + ".enc"

txtpwd = ""
Error:
End Sub

'This one is for Drag and Drop

Private Sub Encryptb_OLEDragDrop(Data As DataObject, Effect As Long, Button As Integer,
Shift As Integer, x As Single, y As Single)
Dim filecounter As Integer

For filecounter = 1 To Data.Files.Count
   If (GetAttr(Data.Files.Item(filecounter)) And vbDirectory) = 0 Then

      olefile = Data.Files(filecounter)
      encrypt = True

      Call check

   End If

Next
End Sub
```

```
Private Sub Decrypt_OLEDragDrop(Data As DataObject, Effect As Long, Button As Integer,
Shift As Integer, x As Single, y As Single)
Dim filecounter As Integer

For filecounter = 1 To Data.Files.Count
   If (GetAttr(Data.Files.Item(filecounter)) And vbDirectory) = 0 Then

      olefile = Data.Files(filecounter)
      encrypt = False
      Call check

   End If

Next
End Sub

Private Sub check()
If encrypt Then
      If txtpwd2.Text = "" Then
       MsgBox "Encrypt private key kosong!!", 0, "Error!"
      txtpwd.SetFocus
      Exit Sub
      Else
      End If

Else

      If txtpwd2.Text = "" Then
      MsgBox "Decrypt private key kosong!!", 0, "Error!"
      txtpwd.SetFocus
      Exit Sub
      End If
End If
```

```
        Call EnDeCryptfile

End Sub

Private Sub Encryptm_Click()
olefile = ""
encrypt = True
Call check
End Sub

Private Sub Decryptm_Click()
olefile = ""
encrypt = False
Call check
End Sub

Private Sub Exitm_Click()
End
End Sub



Private Sub About_Click()
Load frmabout
Call ChangeStay(frmabout, 1)
frmabout.Show
End Sub

'Verify Password
Private Sub Ok_Click()
If temp = txtpwd Then
   Ok.Visible = False
   Cancel.Visible = False
   Call EnDeCryptfile
```

```
Else
   Ok.Visible = False
   Cancel.Visible = False
   form2.Caption = StatusCaption
   txtpwd = ""
   txtpwd.SetFocus
   Exit Sub
End If

form2.Caption = StatusCaption
End Sub

Private Sub Cancel_Click()
Ok.Visible = False
Cancel.Visible = False
form2.Caption = StatusCaption
txtpwd = ""
txtpwd.SetFocus
End Sub
```

```
'-------------------------------------------------------------------------------------------------------------
'                                                file: frmfinish.frm
'                                          created by: Fredy Sadikun Halim
'-------------------------------------------------------------------------------------------------------------
Private Sub Command1_Click()
Unload Me
frmStepTwo.Show
End Sub

Private Sub Form_Load()
Dim p As Integer
Dim q As Integer

p = 43
q = 67
Label1.Caption = "d = " & d
Label2.Caption = "p = " & p
Label3.Caption = "q = " & q
Label4.Caption = "m = " & m
Label5.Caption = "n = " & n
Label6.Caption = "e = " & e

Text1 = n
Text2 = e
Text3 = n
Text4 = d
End Sub
```

```
'-------------------------------------------------------------------------------------------------------------
'                                                file: frmstepthree.frm
'                                          created by: Fredy Sadikun Halim
'-------------------------------------------------------------------------------------------------------------
Dim s
Private Sub Command1_Click()
FindD
End Sub

Private Sub Command2_Click()
If Label7.Caption = "" Then
   MsgBox "calculate d!"
   Exit Sub
End If
d = s - 1
frmFinish.Show
Unload Me
End Sub

'de mod m = 1
Private Sub Form_Load()
Dim p As Integer
Dim q As Integer

p = 43
q = 67

Label2.Caption = "p = " & p
Label3.Caption = "q = " & q
Label4.Caption = "m = " & m
Label5.Caption = "n = " & n
Label6.Caption = "e = " & e
End Sub
```

```
Private Sub FindD()


s = 1
Do While i <> 1
   i = (s * e) Mod m
   s = s + 1
Loop
Label7.Caption = "d = " & s - 1
End Sub

'-------------------------------------------------------------------------------------------------------------------
'                                                file: frmsteptwo.frm
'                                          created by: Fredy Sadikun Halim
'-------------------------------------------------------------------------------------------------------------------
Private Sub Command1_Click()
For t = Text1 To Text2
   a = GCD(t, m)
   If a = 1 Then
      List1.AddItem t
   End If
Next t

End Sub

Private Sub Command2_Click()
If Text3 = "" Then
   MsgBox "Please select an e number!"
   Exit Sub
End If
e = Text3
frmStepThree.Show
Unload Me
End Sub
```

```
Private Sub Form_Load()
Dim p As Integer
Dim q As Integer

p = 43
q = 67

Label1.Caption = "p = " & p
Label2.Caption = "q = " & q
Label3.Caption = "m = " & (p - 1) * (q - 1)
m = (p - 1) * (q - 1)
Label4.Caption = "n = " & p * q
n = p * q
End Sub

Private Function GCD(ByVal i As Long, ByVal j As Long) As Long

Dim k As Long
i = Abs(i)
j = Abs(j)
If i + j = 0 Then Exit Function

Do While i > 0
   k = i
   i = j Mod i
   j = k
Loop
GCD = k
End Function

Private Sub List1_DblClick()
Text3 = List1.List(List1.ListIndex)
End Sub
```

```
'-------------------------------------------------------------------------------------------------------------
'                                              file: misc.bas
'                                        created by: Fredy Sadikun Halim
'-------------------------------------------------------------------------------------------------------------
'For the ini-File
Public Declare Function WritePrivateProfileString Lib "kernel32" Alias
"WritePrivateProfileStringA" (ByVal lpApplicationName As String, ByVal lpKeyName As Any,
ByVal lpString As Any, ByVal lpFileName As String) As Long
Public Declare Function WriterivateProfileSection Lib "kernel32" Alias
"WritePrivateProfileSectionA" (ByVal lpAppName As String, ByVal lpString As String, ByVal
lpFileName As String) As Long
Public Declare Function GetPrivateProfileString Lib "kernel32" Alias
"GetPrivateProfileStringA" (ByVal lpApplicationName As String, ByVal lpKeyName As Any,
ByVal lpDefault As String, ByVal lpReturnedString As String, ByVal nSize As Long, ByVal
lpFileName As String) As Long

'Stay Function
Declare Function SetWindowPos Lib "user32" (ByVal hwnd As Long, ByVal hWndInsertAfter
As Long, ByVal x As Long, ByVal y As Long, ByVal cx As Long, ByVal cy As Long, ByVal
wFlags As Long) As Long

Global Const SWP_NOMOVE = 2
Global Const SWP_NOSIZE = 1
Global Const WndFlags = SWP_NOMOVE Or SWP_NOSIZE
Global Const HWND_TOPMOST = -1
Global Const HWND_NOTOPMOST = -2
Global Const conSwpNoActivate = &H10
Global Const conSwpShowWindow = &H40

'For Fileselection
Public Type OPENFILENAME
    lStructSize As Long
    hwndOwner As Long
    hInstance As Long
```

```
    lpstrFilter As String
    lpstrCustomFilter As String
    nMaxCustFilter As Long
    nFilterIndex As Long
    lpstrFile As String
    nMaxFile As Long
    lpstrFileTitle As String
    nMaxFileTitle As Long
    lpstrInitialDir As String
    lpstrTitle As String
    flags As Long
    nFileOffset As Integer
    nFileExtension As Integer
    lpstrDefExt As String
    lCustData As Long
    lpfnHook As Long
    lpTemplateName As String
End Type

Const OFN_READONLY = &H1
Const OFN_OVERWRITEPROMPT = &H2
Const OFN_HIDEREADONLY = &H4
Const OFN_NOCHANGEDIR = &H8
Const OFN_SHOWHELP = &H10
Const OFN_ENABLEHOOK = &H20
Const OFN_ENABLETEMPLATE = &H40
Const OFN_ENABLETEMPLATEHANDLE = &H80
Const OFN_NOVALIDATE = &H100
Const OFN_ALLOWMULTISELECT = &H200
Const OFN_EXTENSIONDIFFERENT = &H400
Const OFN_PATHMUSTEXIST = &H800
Const OFN_FILEMUSTEXIST = &H1000
Const OFN_CREATEPROMPT = &H2000
Const OFN_SHAREAWARE = &H4000
```

```
Const OFN_NOREADONLYRETURN = &H8000
Const OFN_NOTESTFILECREATE = &H10000
Const OFN_NONETWORKBUTTON = &H20000
Const OFN_NOLONGNAMES = &H40000 ' force no long names for 4.x modules
Const OFN_EXPLORER = &H80000 ' new look commdlg
Const OFN_NODEREFERENCELINKS = &H100000
Const OFN_LONGNAMES = &H200000 ' force long names for 3.x modules
Const OFN_SHAREFALLTHROUGH = 2
Const OFN_SHARENOWARN = 1
Const OFN_SHAREWARN = 0

Public Declare Function GetSaveFileName Lib "comdlg32" Alias "GetSaveFileNameA"
(pOpenfilename As OPENFILENAME) As Long

'For the Test Speed-Routine
Public Declare Function GetTickCount Lib "kernel32" () As Long


'Public Dim's
Public Splash As Boolean
Public Stay As Boolean
Public Verify As Boolean
Public Winzippath As String
Public Winzip As Boolean
Public oldzipname As String
'Hash Public's
Public Hash_Bo As Boolean
Public Hash_Sel As String


Public Function SaveDialog(Form1 As Form, Filter As String, Title As String, InitDir As String)
As String
Dim ofn As OPENFILENAME
Dim a As Long
```

```
ofn.lStructSize = Len(ofn)
ofn.hwndOwner = Form1.hwnd
ofn.hInstance = App.hInstance
If Right$(Filter, 1) <> "|" Then Filter = Filter & "|"
For a = 1 To Len(Filter)
If Mid$(Filter, a, 1) = "|" Then Mid$(Filter, a, 1) = Chr$(0)
Next
ofn.lpstrFilter = Filter
ofn.lpstrFile = Space$(254)
ofn.nMaxFile = 255
ofn.lpstrFileTitle = Space$(254)
ofn.nMaxFileTitle = 255
ofn.lpstrInitialDir = InitDir
ofn.lpstrTitle = Title
ofn.flags = OFN_HIDEREADONLY Or OFN_CREATEPROMPT
a = GetSaveFileName(ofn)
If (a) Then
SaveDialog = Left$(Trim$(ofn.lpstrFile), Len(Trim$(ofn.lpstrFile)) - 1)
Else
SaveDialog = ""
End If
End Function

Sub Writetoini(ByVal datei As String, ByVal sektion As String, ByVal eintrag As String, ByVal
wert As String)
   Dim res As Integer
   res = WritePrivateProfileString(sektion, eintrag, wert, datei)
End Sub

Public Function readfromini(ByVal datei As String, ByVal sektion As String, ByVal eintrag As
String, ByVal default As String) As String
   Dim RetVal As String, res As Integer
   RetVal = String$(255, 0)
   res = GetPrivateProfileString(sektion, eintrag, default, RetVal, Len(RetVal), datei)
```

```
    readfromini = Left(RetVal, res)
End Function

'************************************
'Checks if a file exists
'************************************

Function FileExist(File As String) As Boolean

Dim iFHandle As Integer
On Local Error GoTo Isnich

FileExist = True
iFHandle = FreeFile
Open File For Input As #iFHandle
Close #iFHandle
Exit Function

Isnich:
FileExist = False
Resume Next
End Function


Public Sub Str2ByteArray(StringIn As String, ByteArray() As Byte)
    ByteArray = StrConv(StringIn, vbFromUnicode)
End Sub

Public Sub ChangeStay(frm As Form, stayc As Boolean)
Dim tempst As Integer

If stayc Then
```

```
   tempst = SetWindowPos(frm.hwnd, HWND_TOPMOST, frm.Left \ Screen.TwipsPerPixelX,
frm.Top \ Screen.TwipsPerPixelY, frm.Width \ Screen.TwipsPerPixelX, frm.Height \
Screen.TwipsPerPixelY, 0)
Else
   tempst = SetWindowPos(frm.hwnd, HWND_NOTOPMOST, frm.Left \
Screen.TwipsPerPixelX, frm.Top \ Screen.TwipsPerPixelY, frm.Width \
Screen.TwipsPerPixelX, frm.Height \ Screen.TwipsPerPixelY, 0)
End If
End Sub

'-----------------------------------------------------------------------------------------------------------
'                                          file: moddeclares.bas
'                                  created by: Fredy Sadikun Halim
'-----------------------------------------------------------------------------------------------------------
Public Declare Function GlobalAlloc Lib "kernel32" (ByVal wFlags As Long, ByVal dwBytes
As Long) As Long
Public Declare Function GlobalFree Lib "kernel32" (ByVal hMem As Long) As Long
Public Declare Function GlobalLock Lib "kernel32" (ByVal hMem As Long) As Long
Public Declare Function GlobalUnlock Lib "kernel32" (ByVal hMem As Long) As Long

Public Declare Sub RtlMoveMemory Lib "kernel32" (Destination As Any, Source As Any,
ByVal Length As Long)

'-----------------------------------------------------------------------------------------------------------
'                                          file: modgeneral.bas
'                                  created by: Fredy Sadikun Halim
'-----------------------------------------------------------------------------------------------------------
Public p, q, n, m, e, d
```

```
'-------------------------------------------------------------------------------------------------------------
'                                              file: modmain.bas
'                                        created by: Fredy Sadikun Halim
'-------------------------------------------------------------------------------------------------------------
Private BitwiseOperator As New clsBitwiseOperator
Private lngCRC32Table(255) As Long

Public Function InitializeCRC32Table(Optional ByVal lngPolynomial As Long =
&HEDB88320, Optional ByVal lngInitializeValue As Long = &HFFFFFFFF) As Long
   Dim lngBytes As Long, lngBits As Long
   Dim lngCRC32 As Long, lngTemporaryCRC32 As Long

   '// cycle through every byte available and generate a lookup table
   For lngBytes = 0 To 255

      lngCRC32 = lngBytes

      For lngBits = 0 To 7
         lngTemporaryCRC32 = BitwiseOperator.ShiftRight(lngCRC32, 1)

         If (lngCRC32 And &H1) Then
            lngCRC32 = lngTemporaryCRC32 Xor lngPolynomial
         Else
            lngCRC32 = lngTemporaryCRC32
         End If
      Next lngBits

      lngCRC32Table(lngBytes) = lngCRC32
   Next lngBytes

   InitializeCRC32Table = lngInitializeValue
End Function
```

```
Public Function GenerateCRC32(ByVal strBuffer As String, ByVal lngInitializeValue As Long)
As Long
   Dim lngBytes As Long, bytCurrentByte As Byte
   Dim lngActualValue As Long
   Dim lngTableValue As Long, lngIndex As Long

   For lngBytes = 1 To Len(strBuffer)

      bytCurrentByte = Asc(Mid$(strBuffer, lngBytes, 1))

      lngActualValue = BitwiseOperator.ShiftRight(lngInitializeValue, 8)

      lngIndex = lngInitializeValue And &HFF
      lngIndex = lngIndex Xor bytCurrentByte
      lngTableValue = lngCRC32Table(lngIndex)

      lngInitializeValue = lngActualValue Xor lngTableValue
   Next lngBytes

   GenerateCRC32 = lngInitializeValue Xor &HFFFFFFFF
End Function

'-------------------------------------------------------------------------------------------------------------
'                                        file: clsbitwiseoperator.cls
'                                    created by: Fredy Sadikun Halim
'-------------------------------------------------------------------------------------------------------------
Option Explicit

Private Type tPD
   hMem              As Long
   PtrToOldCode      As Long
End Type
Private ProcDetails()   As tPD
```

```
Private VTIndex         As Long
Private Code            As Byte
Private CodeSize        As Long
Private PtrToNewCode    As Long
Private PtrToMyself     As Long
Private i               As Long

Private Sub Class_Initialize()
   VTIndex = -1    'initialize index into Virtual Table
   ShiftLeft 0, 0  'this sets up m/c code and modifies the VT
   ShiftRight 0, 0  'this sets up m/c code and modifies the VT
   RotateLeft 0, 0  'this sets up m/c code and modifies the VT
   RotateRight 0, 0  'this sets up m/c code and modifies the VT
End Sub


Public Function ShiftLeft(ByVal lngValue As Long, ByVal lngShift As Long) As Long

  'this is in fact only called once during class initialize
  'subsequent calls are diverted (via the VT) to the m/c code

   DivertTo "8B442408 8B4C240C 8B542410 D3E0 8902 31C0 C21000"

End Function

Public Function ShiftRight(ByVal lngValue As Long, ByVal lngShift As Long) As Long

  'this is in fact only called once during class initialize
  'subsequent calls are diverted (via the VT) to the m/c code

   DivertTo "8B442408 8B4C240C 8B542410 D3E8 8902 31C0 C21000"

End Function
```

```
Public Function RotateLeft(ByVal lngValue As Long, ByVal lngRotate As Long) As Long

  'this is in fact only called once during class initialize
  'subsequent calls are diverted (via the VT) to the m/c code

    DivertTo "8B442408 8B4C240C 8B542410 D3C0 8902 31C0 C21000"

End Function

Public Function RotateRight(ByVal lngValue As Long, ByVal lngRotate As Long) As Long

  'this is in fact only called once during class initialize
  'subsequent calls are diverted (via the VT) to the m/c code

    DivertTo "8B442408 8B4C240C 8B542410 D3C8 8902 31C0 C21000"

End Function

Private Sub DivertTo(ByVal HexCode As String)

    VTIndex = VTIndex + 1 'inc index into VT
    ReDim Preserve ProcDetails(0 To VTIndex) 'adjust array size

    HexCode = Replace$(HexCode, " ", "") 'remove spaces from hex code
    CodeSize = Len(HexCode) / 2 'length of the resulting binary code (2 hex chars per byte of
code)

    With ProcDetails(VTIndex)
        .hMem = GlobalAlloc(0, CodeSize) 'get memory for m/c code and save handle
        PtrToNewCode = GlobalLock(.hMem) 'get far pointer to allocated memory

        For i = 0 To CodeSize - 1
            Code = Val("&H" & Mid$(HexCode, i + i + 1, 2)) 'convert hex to binary m/c code
            RtlMoveMemory ByVal PtrToNewCode + i, Code, 1 'store it in allocated memory
```

```
        Next i

        .PtrToOldCode = VirtualTableEntry 'save old VT entry; VTIndex determines which entry
        VirtualTableEntry = PtrToNewCode 'overwrite VT entry; VTIndex determines which entry
        GlobalUnlock .hMem 'unlock memory
    End With 'PROCDETAILS(VTINDEX)

End Sub

Private Property Let VirtualTableEntry(ByVal FarPointer As Long)

    RtlMoveMemory PtrToMyself, ByVal ObjPtr(Me), 4 'get pointer to object (Me)
    RtlMoveMemory ByVal PtrToMyself + &H1C + VTIndex * 4, FarPointer, 4 'put VT entry

End Property

Private Property Get VirtualTableEntry() As Long

    RtlMoveMemory PtrToMyself, ByVal ObjPtr(Me), 4 'get pointer to object (Me)
    RtlMoveMemory VirtualTableEntry, ByVal PtrToMyself + &H1C + VTIndex * 4, 4 'get VT
entry

End Property

Private Sub Class_Terminate()

    For VTIndex = VTIndex To 0 Step -1 'VTIndex still points to the last VT entry overwritten
        With ProcDetails(VTIndex)
            VirtualTableEntry = .PtrToOldCode 'restore VT entry; VTIndex determines which entry
            GlobalFree .hMem 'release memory used for m/c code
        End With 'PROCDETAILS(VTINDEX)
    Next VTIndex

End
```

```
'-------------------------------------------------------------------------------------------------------------
'                                              file: RSA.bas
'                                        created by: Fredy Sadikun Halim
'-------------------------------------------------------------------------------------------------------------
Public p As Double, q As Double
Public PHI As Double
Public Function Mult(ByVal x As Double, ByVal pg As Double, ByVal m As Double) As
Double
On Error GoTo error1
y = 1
   Do While pg > 0
      Do While (pg / 2) = Int((pg / 2))
         x = nMod((x * x), m)
         pg = pg / 2
      Loop
      y = nMod((x * y), m)
      pg = pg - 1
   Loop
   Mult = y
   Exit Function
error1:
y = 0
End Function

Private Function nMod(x As Double, y As Double) As Double
  On Error Resume Next
  Dim z#
  z = x - (Int(x / y) * y)
  nMod = z
End Function
Public Function enc(ByVal tIp As String, eE As Double, eN As Double) As String
On Error Resume Next
Dim encSt As String
encSt = ""
```

```
e2st = ""
   If tIp = "" Then Exit Function
   For i = 1 To Len(tIp)
      encSt = encSt & Mult(CLng(Asc(Mid(tIp, i, 1))), eE, eN) & "+"
   Next i
enc = encSt
End Function

Public Function dec(ByVal tIp As String, dD As Double, dN As Double) As String
On Error Resume Next
Dim decSt As String
decSt = ""
For z = 1 To Len(tIp)
   ptr = InStr(z, tIp, "+")
   tok = Val(Mid(tIp, z, ptr))
   decSt = decSt + Chr(Mult(tok, dD, dN))
   z = ptr
Next z
dec = decSt
End Function

'-------------------------------------------------------------------------------------------------------------
'                                                file: SHA.bas
'                                        created by: Fredy Sadikun Halim
'-------------------------------------------------------------------------------------------------------------
Public bufA As Double
Public bufB As Double
Public bufC As Double
Public bufD As Double
Public bufE As Double

Public sign_msg As String

Private Function NotX(ByVal ip As Double) As Double
```

```
    NotX = UpBound - ip - 1
End Function

Private Function XorY(ByVal x As Double, ByVal y As Double) As Double
    Dim x1 As Double
    Dim x2 As Double
    Dim x3 As Double
    Dim x4 As Double
    x2 = xmody(x, 2 ^ 16)
    x1 = Int(x / (2 ^ 16))
    Y2 = xmody(y, 2 ^ 16)
    x1 = Int(y / (2 ^ 16))
    XorY = (x1 Or Y1) * (2 ^ 16) + (x2 Or Y2)
End Function

Private Function XandY(ByVal x As Double, ByVal y As Double)
    Dim x1 As Double
    Dim x2 As Double
    Dim x3 As Double
    Dim x4 As Double
    x2 = xmody(x, 2 ^ 16)
    x1 = Int(x / (2 ^ 16))
    Y2 = xmody(y, 2 ^ 16)
    x1 = Int(y / (2 ^ 16))
    XandY = (x1 And Y1) * (2 ^ 16) + (x2 And Y2)
End Function

Private Function XxorY(ByVal x As Double, ByVal y As Double)
    Dim x1 As Double
    Dim x2 As Double
    Dim x3 As Double
    Dim x4 As Double
    x2 = xmody(x, 2 ^ 16)
    x1 = Int(x / (2 ^ 16))
```

```
    Y2 = xmody(y, 2 ^ 16)
    x1 = Int(y / (2 ^ 16))
    XxorY = (x1 Xor Y1) * (2 ^ 16) + (x2 Xor Y2)
End Function



Public Sub init_sha_buffer()
    bufA = 1732584193
    bufB = 4023233417#
    bufC = 2562383102#
    bufD = 271733878
    bufE = 3285377520#
End Sub

Public Function shapadding(msg As String) As String
    msg = msg & Chr(128)
    Do
        msg = msg & Chr(0)
    Loop Until LenB(msg) Mod 64 = 56
    shapadding = msg
End Function

Public Function append_length(msg As String, org_length As Double) As String
    Dim append_str As String
    Dim i As Integer
    append_str = ""

    For i = 1 To 8
        If org_length <> 0 Then
            append_str = Chr(xmody(org_length, 100)) & append_str
        Else
            append_str = Chr(0) & append_str
        End If
```

```
        org_length = Int(org_length / 100)
    Next i
    append_length = msg & append_str
End Function

Public Sub sha_hash_msg(msg As String, k As Integer)
    Dim org_length As Double
    Dim block(1 To 64) As Integer
    Dim i, j, y As Double
    Dim u As Integer
    org_length = LenB(msg)
    msg = shapadding(msg)
    msg = append_length(msg, org_length)
    Call init_sha_buffer
    For i = 0 To LenB(msg) \ 64 - 1
        For j = 1 To 64
            block(j) = AscB(MidB(msg, i * 64 + j, 1))
        Next j
        Call H_sha(block(), k)
    Next i
End Sub

Public Sub H_sha(block() As Integer, q As Integer)

    Dim org_A, org_B, org_C, org_D, org_E As Double
    Dim tempA, tempB, tempC, tempD, tempE As Double
    Dim Wt(80), Kt As Double
    Dim i, t, j As Integer
    Dim limit As Double

    limit = 2 ^ 32
    org_A = bufA
    org_B = bufB
    org_C = bufC
```

```
org_D = bufD
org_E = bufE

Wt(0) = q Mod 5

For i = 1 To 15
   Wt(i) = 0
   Wt(i) = Wt(i) + block(i * 4 + 1)
   Wt(i) = Wt(i) + block(i * 4)
Next i


For t = 0 To 79
   If 0 <= t <= 19 Then Kt = 1518500249
   If 20 <= t <= 39 Then Kt = 1859775393
   If 40 <= t <= 59 Then Kt = 2400959708#
   If 60 <= t <= 79 Then Kt = 3395469782#
   If t >= 16 Then Wt(t) = XxorY(XxorY(XxorY(Wt(t - 16), Wt(t - 14)), Wt(t - 8)), Wt(t - 3))
   tempA = xmody((CLS(bufA, 5) + Ft(bufB, bufC, bufD, t)), limit)
   tempA = xmody((tempA + bufE), limit)
   tempA = xmody((tempA + Wt(t)), limit)
   tempA = xmody((tempA + Kt), limit)
   tempB = bufA
   tempC = CLS(bufB, 30)
   tempD = bufC
   tempE = bufD
   bufA = tempA
   bufB = tempB
   bufC = tempC
   bufD = tempD
   bufE = tempE
Next t
bufA = xmody((bufA + org_A), limit)
bufB = xmody((bufB + org_B), limit)
```

```
    bufC = xmody((bufC + org_C), limit)
    bufD = xmody((bufD + org_D), limit)
    bufE = xmody((bufE + org_E), limit)
End Sub

Public Function Ft(ByVal b As Double, ByVal c As Double, ByVal d As Double, ByVal t As
Integer) As Double
Dim r As Double

    If 0 <= t <= 19 Then
        r = XorY(XandY(b, c), XandY(NotX(b), d))
    End If
    If 20 <= t <= 39 Then
        r = XxorY(XxorY(b, c), d)
    End If
    If 40 <= t <= 59 Then
        r = XorY(XorY(XandY(b, c), XandY(b, d)), XandY(c, d))
    End If
    If 60 <= t <= 79 Then
        r = XxorY(XxorY(b, c), d)
    End If
    Ft = r
End Function

Public Function CLS(ByVal a As Double, ByVal s As Integer) As Double
    Dim lbits As Double
    Dim rbits As Double
    Dim ss As Integer
    s = xmody(s, 32)
    ss = 32 - s
    lbits = Int(x / (2 ^ ss))
    rbits = xmody(a, 2 ^ ss)
    CLS = xmody(rbits * (2 ^ s), 2 ^ 32) + lbits
    CLS = xmody(CLS, 2 ^ 32)
```

```
End Function

Public Function xmody(ByVal x As Double, ByVal y As Double) As Double
    Dim q As Double
    q = Int(x / y)
    x = x - q * y
    Do
        If x >= y Then x = x - y
    Loop Until x < y
    xmody = x
End Function
```