# LAMPIRAN A

# LISTING PROGRAM

```
Public Sub Calculate_Entropy(ByteArray() As Byte)

  Dim Entropy As Double

  Dim CharCount(255) As Long

  Dim X As Long

  Dim Prob As Double

  Dim PacLen As Double

  Dim TotFreq As Long

  TotFreq = UBound(ByteArray) + 1

  Entropy = 0

  For X = 0 To UBound(ByteArray)

    CharCount(ByteArray(X)) = CharCount(ByteArray(X)) + 1

  Next

  PacLen = 0

  For X = 0 To 255

    PacLen = PacLen - CharCount(X) * Log2(CharCount(X) / TotFreq)

  Next

  Entropy = PacLen / TotFreq

  PacLen = CLng(PacLen / 8)

  MsgBox "Entropy = " & Format(Entropy, "0.00") & " bits per byte" & Chr(13) & "Minimum filelengt = " & PacLen & " bytes"

End Sub


Public Function Log2(P)

  If P = 0 Then

    Log2 = 0

  Else

    Log2 = Log(P) / Log(2)

  End If

End Function
```

**This module calculates the CRC32-checksum of a certain bytestream**

```
Dim CrcTableInit As Boolean

Dim CRCTable(0 To 255) As Long


Public Function calcCRC32(ByteArray() As Byte) As Long

  Dim i As Long

  Dim crc As Long

  If CrcTableInit = False Then Call Init_CRCTable

  crc = -1

  For i = 0 To UBound(ByteArray) - 1

    crc = (((crc And &HFFFFFF00) \ &H100) And &HFFFFFF) Xor (CRCTable((crc And &HFF) Xor ByteArray(i)))

  Next i

  crc = crc Xor &HFFFFFFFF

  calcCRC32 = crc

End Function


Private Sub Init_CRCTable()

  Dim i As Long

  Dim J As Long

  Dim Limit As Long

  Dim crc As Long

  Limit = &HEDB88320

  For i = 0 To 255

    crc = i

    For J = 0 To 7

      If crc And 1 Then

        crc = (((crc And &HFFFFFFFE) \ 2) And &H7FFFFFFF) Xor Limit

      Else

        crc = ((crc And &HFFFFFFFE) \ 2) And &H7FFFFFFF

      End If

    Next J

    CRCTable(i) = crc

  Next i

  CrcTableInit = True

End Sub
```

## Location Based Encoding

```
Private Dictionary As String

Private CharCount(256) As Long
```

```
Private Bitlen(255) As Long

Private BitStart() As Integer

Private CharVal(255) As Long

Private OutStream() As Byte

Private OutPos As Long

Private OutByteBuf As Integer

Private OutBitCount As Integer

Private MinBitsToRead As Byte



Public Sub Compress_LBE(ByteArray() As Byte, Optional WichType As Integer = 2)

  Dim AscPos As Byte

  Dim InPos As Long

  Dim OrgFileLenght As Long

  Call Init_LBE

      Call Init_LBE_Flat

  InPos = 0

  OrgFileLenght = UBound(ByteArray)

  ReDim OutStream(500)

  Do While InPos <= UBound(ByteArray)

    AscPos = InStr(Dictionary, Chr(ByteArray(InPos))) - 1

    Call AddBitsToOutStream(CharVal(AscPos), CInt(Bitlen(AscPos)))

    Call update_Model(ByteArray(InPos))

    InPos = InPos + 1

  Loop

'fill up the last byte

  Do While OutBitCount > 0

    Call AddBitsToOutStream(0, 1)

  Loop

  ReDim ByteArray(OutPos + 3)

  Call CopyMem(ByteArray(0), OutStream(0), OutPos)

  ByteArray(OutPos) = Int(OrgFileLenght / &H1000000) And &HFF

  ByteArray(OutPos + 1) = Int(OrgFileLenght / &H10000) And &HFF

  ByteArray(OutPos + 2) = Int(OrgFileLenght / &H100) And &HFF

  ByteArray(OutPos + 3) = OrgFileLenght And &HFF

End Sub



Public Sub DeCompress_LBE(ByteArray() As Byte, Optional WichType As Integer = 2)

  Dim InPos As Long

  Dim InBit As Integer

  Dim OrgFileLenght As Long

  Dim Blen As Byte

  Dim BitVal As Long

  Dim AscVal As Byte

  Dim X As Long

  Call Init_LBE

      Call Init_LBE_Flat

  OrgFileLenght = ByteArray(UBound(ByteArray) - 3)

  OrgFileLenght = CLng(OrgFileLenght) * 256 + ByteArray(UBound(ByteArray) - 2)

  OrgFileLenght = CLng(OrgFileLenght) * 256 + ByteArray(UBound(ByteArray) - 1)

  OrgFileLenght = CLng(OrgFileLenght) * 256 + ByteArray(UBound(ByteArray))

  InPos = 0

  InBit = 0

  ReDim OutStream(OrgFileLenght)

  Do While OutPos <= OrgFileLenght

    BitVal = 0

    Blen = 0

    For X = 1 To MinBitsToRead

      BitVal = BitVal * 2 + ReadBitsFromArray(ByteArray, InPos, InBit, 1)

      Blen = Blen + 1

    Next
```

```
    Do
        For X = BitStart(Blen) To BitStart(Blen + 1) - 1
            If CharVal(X) = BitVal Then
                AscVal = ASC(Mid(Dictionary, X + 1, 1))
                Call AddCharToArray(OutStream, OutPos, AscVal)
                Call update_Model(AscVal)
                Exit Do
            End If
        Next
        BitVal = BitVal * 2 + ReadBitsFromArray(ByteArray, InPos, InBit, 1)
        Blen = Blen + 1
    Loop
  Loop
  ReDim ByteArray(OutPos - 1)
  Call CopyMem(ByteArray(0), OutStream(0), OutPos)
End Sub

Private Sub Init_LBE()
  Dim X As Integer
  Dictionary = ""
  For X = 0 To 255
    Dictionary = Dictionary & Chr(X)
    CharCount(X) = 0
  Next
  CharCount(256) = 0
  Dim OutStream(500)
  OutPos = 0
  OutByteBuf = 0
  OutBitCount = 0
End Sub

Private Sub update_Model(Char As Byte)
  Dim Dictpos As Integer
  Dim OldPos As Integer
  Dim Temp As Long
  Dictpos = InStr(Dictionary, Chr(Char)) - 1
  OldPos = Dictpos
  CharCount(Dictpos) = CharCount(Dictpos) + 1
  Do While Dictpos > 0
    If CharCount(Dictpos) < CharCount(Dictpos - 1) Then Exit Do
    Temp = CharCount(Dictpos - 1)
    CharCount(Dictpos - 1) = CharCount(Dictpos)
    CharCount(Dictpos) = Temp
    Dictpos = Dictpos - 1
  Loop
  If OldPos = Dictpos Then Exit Sub
  Dictionary = Left(Dictionary, Dictpos) & Chr(Char) & Mid(Dictionary, Dictpos + 1, OldPos - Dictpos) & Mid(Dictionary, OldPos + 2)
End Sub

Private Sub Init_LBE_Flat()
'
' 0  1  5  9  14  20  27
' 2  4  8  13  19  26
' 3  7  12  18  25
' 6  11  17  24
' 10  16  23
' 15  22
' 21
'  a first 1 represents that the row has reached
'  the second 1 represents that the character has been reached
'
' 0 = "11"     10= "000011"   20= "1000001"
' 1 = "011"    11= "000101"
' 2 = "101"    12= "001001"
' 3 = "0011"   13= "010001"
```

```
' 4 = "0101"     14= "100001"

' 5 = "1001"     15= "0000011"

' 6 = "00011"    16= "0000101"

' 7 = "00101"    17= "0001001"

' 8 = "01001"    18= "0010001"

' 9 = "10001"    19= "0100001"

' There are 20 characters that will be profitable

    Dim X As Integer

    Dim Value As Long

    Dim Blen As Byte

    MinBitsToRead = 2

    Value = 3

    Blen = 1

    ReDim BitStart(10)

    For X = 0 To 255

      If Value > 2 ^ Blen Then

        Blen = Blen + 1

        Value = 3

        If Blen > UBound(BitStart) Then

          ReDim Preserve BitStart(Blen)

        End If

        BitStart(Blen) = X

      End If

      Bitlen(X) = Blen

      CharVal(X) = Value

      Value = (Value * 2) - 1

    Next

    ReDim Preserve BitStart(Blen + 1)

    BitStart(Blen + 1) = 256

End Sub

Private Function BinToDec(BinValue As String)

    Dim X As Integer

    For X = 1 To Len(BinValue)

      If Mid(BinValue, X, 1) = "1" Then BinToDec = BinToDec + 2 ^ (Len(BinValue) - X)

    Next

End Function

this sub will add an amount of bits into the outputstream

Private Sub AddBitsToOutStream(Number As Long, Numbits As Integer)

    Dim X As Long

    For X = Numbits - 1 To 0 Step -1

      OutByteBuf = OutByteBuf * 2 + (-1 * ((Number And CDbl(2 ^ X)) > 0))

      OutBitCount = OutBitCount + 1

      If OutBitCount = 8 Then

        OutStream(OutPos) = OutByteBuf

        OutBitCount = 0

        OutByteBuf = 0

        OutPos = OutPos + 1

        If OutPos > UBound(OutStream) Then

          ReDim Preserve OutStream(OutPos + 500)

        End If

      End If

    Next

End Sub


'this function will return a value out of the amaunt of bits you asked for

Private Function ReadBitsFromArray(FromArray() As Byte, FromPos As Long, FromBit As Integer, Numbits As Integer) As Long

    Dim X As Integer

    Dim Temp As Long

    For X = 1 To Numbits

      Temp = Temp * 2 + (-1 * ((FromArray(FromPos) And 2 ^ (7 - FromBit)) > 0))

      FromBit = FromBit + 1

      If FromBit = 8 Then

        If FromPos + 1 > UBound(FromArray) Then

          Do While X < Numbits
```

```
            Temp = Temp * 2
            X = X + 1
          Loop
          FromPos = FromPos + 1
          Exit For
        End If
        FromPos = FromPos + 1
        FromBit = 0
      End If
    Next
    ReadBitsFromArray = Temp
End Function


'this sub will add a char into the outputstream
Private Sub AddCharToArray(Toarray() As Byte, ToPos As Long, Char As Byte)
    If ToPos > UBound(Toarray) Then ReDim Preserve Toarray(ToPos + 500)
    Toarray(ToPos) = Char
    ToPos = ToPos + 1
End Sub
```

**Prekompresi**

**Frekuensi shifter**

```
Option Explicit

Private Dictionary As String
Private CharCount(256) As Long

Public Sub FrequentShifter_Coder(ByteArray() As Byte)
    Dim X As Long
    Dim Temp As Byte
    Call Init_FrequentShifter
    For X = 0 To UBound(ByteArray)
      Temp = ByteArray(X)
      ByteArray(X) = InStr(Dictionary, Chr(Temp)) - 1
      'Debug.Print X, Temp, ByteArray(X)'
      Call update_Model(Temp)
    Next
    'For X = 0 To 255'
    ' Debug.Print X; "="; Mid(Dictionary, X + 1, 1); " , ";'

Public Sub FrequentShifter_DeCoder(ByteArray() As Byte)
    Dim X, XX As Long
    Dim Temp As Byte
    Call Init_FrequentShifter
    For X = 0 To UBound(ByteArray)
      Temp = ASC(Mid(Dictionary, ByteArray(X) + 1, 1))
      ByteArray(X) = Temp
      Call update_Model(Temp)
Mid(Dictionary, XX + 1, 1); ",";'
    Next
End Sub
Private Sub Init_FrequentShifter()
    Dim X As Integer
    Dictionary = ""
    For X = 0 To 255
      Dictionary = Dictionary & Chr(X)
      CharCount(X) = 0
    Next
    CharCount(256) = 0
End Sub

Private Sub update_Model(Char As Byte)
    Dim Dictpos As Integer
    Dim OldPos As Integer
    Dim Temp As Long
    Dictpos = InStr(Dictionary, Chr(Char)) - 1
    OldPos = Dictpos
```

```
    CharCount(Dictpos) = CharCount(Dictpos) + 1

    Do While Dictpos > 0

      If CharCount(Dictpos) < CharCount(Dictpos - 1) Then Exit Do

      Temp = CharCount(Dictpos - 1)

      CharCount(Dictpos - 1) = CharCount(Dictpos)

      CharCount(Dictpos) = Temp

      Dictpos = Dictpos - 1

    Loop

    If OldPos = Dictpos Then Exit Sub

    Dictionary = Left(Dictionary, Dictpos) & Chr(Char) & Mid(Dictionary, Dictpos + 1, OldPos - Dictpos) & Mid(Dictionary, OldPos + 2)

End Sub
```

## Burrows Wheeler Transform

```
'This is a Burrows-Wheeler transform coder

'It works by sorting al the data in lexicographical order

'and the it takes the last character of each array

'---------- Transform----------

'The array must be seen as a circle so LAST+1=FIRST

'then you make copies of it len(text) times but every copy has shift 1 to the right

'then you can sort the strings

'

'example: Hariyanto

'

' original                sorted

'

' 0 = H a r i y a n t o     0 = 0 = H a r i y a n t o

' 1 = a r i y a n t o H     1 = 5 = a n t o H a r i y

' 2 = r i y a n t o H a     2 = 1 = a r i y a n t o H

' 3 = i y a n t o H a r     3 = 3 = i y a n t o H a r

' 4 = y a n t o H a r i     4 = 6 = n t o H a r i y a

' 5 = a n t o H a r i y     5 = 8 = o H a r i y a n t

' 6 = n t o H a r i y a     6 = 2 = r i y a n t o H a

' 7 = t o H a r i y a n     7 = 7 = t o H a r i y a n

' 8 = o H a r i y a n t     8 = 4 = y a n t o H a r i

'

'if u take the last characters of the sorted strings u'll get

' oyHratani with prefix 2

'don't forget the prefix (without it you won't get the original text back)

'---------- Decode----------

'The thing we need to do is create another string with the same contents

'and sort that other string so that we get

'

'place: 0 1 2 3 4 5 6 7 8

'org:  o y H r a t a n i

'new:  H a a i n o r t y

'

'now where gone create a transformation table

'If we take the first 'H' as position 0 and look it up in the org we'll see

'that we find the first 'H' in place 2. This means that TV(0)=2

'The first 'a' in new we'll find as the first 'a' in org. so TV(1)=4

'after doing all the characters u will get a table like this

'( 2 , 4 , 6 , 8 , 7 , 0 , 3 , 5 , 1 ) this is base 0

'with help if the prefix we now gen get the original string back according to

'the next

'

'  Offset=prefix
```

```
' For i = 0 To lenght of text

'               BWT_DeCodecString = BWT_DeCodecString & Chr(L(Offset))

'    Offset = TV(Offset)

' Next

'



Public Sub BWT_CodecArray4(ByteArray() As Byte)

  Dim F() As Long

  Dim FTemp() As Long

  Dim OutStream() As Byte

  Dim i As Long

  Dim J As Long

  Dim b As Long

  Dim L As Long

  Dim t As Long

  Dim r As Long

  Dim d As Long

  Dim K As Long

  Dim Y As Long

  Dim Z As Long

  Dim Q As Integer

  Dim ASC As Integer

  Dim FileLength As Long

  Dim P(1 To 100) As Long

  Dim W(1 To 100) As Long

  Dim X As Long

  Dim Prefix As Long

  Dim CharCount(255) As Long

  Dim TempCount() As Long

  Dim Spos(255) As Long

  Dim TPos() As Long

  Dim CheckPos As Long

  Dim NuPos As Long

  FileLength = UBound(ByteArray)

  ReDim F(FileLength)

'This is the speedsort method wich is the fastest as far as i know

'first whe collect the frequentie of each char

  For X = 0 To FileLength

    CharCount(ByteArray(X)) = CharCount(ByteArray(X)) + 1

  Next

'then where gone create the offset pointers

  NuPos = 0

  For X = 0 To 255

    If CharCount(X) > 0 Then

      Spos(X) = NuPos

      NuPos = NuPos + CharCount(X)

    End If

  Next

'and last where place the pointers in order

  For X = 0 To FileLength

    F(Spos(ByteArray(X))) = X

    Spos(ByteArray(X)) = Spos(ByteArray(X)) + 1

  Next



'The BytePointers are now sorted

'and now where cone try to create lexicograpical sorted arrays

'Lets start with a speedsort method and finish the job with Quicksort

  For ASC = 0 To 255

    If CharCount(ASC) > 1 Then

      ReDim TempCount(255)

      ReDim TPos(255)

      ReDim FTemp(CharCount(ASC) - 1)

      NuPos = Spos(ASC) - CharCount(ASC)

      Z = 0

      For X = NuPos To NuPos + CharCount(ASC) - 1

        FTemp(Z) = F(X)

        Z = Z + 1
```

```
        Next

        For X = 0 To CharCount(ASC) - 1

          Z = FTemp(X) + 1: If Z > FileLength Then Z = Z - FileLength - 1

          TempCount(ByteArray(Z)) = TempCount(ByteArray(Z)) + 1

        Next

        For X = 0 To 255

          If TempCount(X) > 0 Then

            TPos(X) = NuPos

            NuPos = NuPos + TempCount(X)

          End If

        Next

        For X = 0 To CharCount(ASC) - 1

          Z = FTemp(X) + 1: If Z > FileLength Then Z = Z - FileLength - 1

          F(TPos(ByteArray(Z))) = FTemp(X)

          TPos(ByteArray(Z)) = TPos(ByteArray(Z)) + 1

        Next

        NuPos = Spos(ASC) - CharCount(ASC)

        For Q = 0 To 255

          If TempCount(Q) > 0 Then

            L = NuPos

            r = NuPos + TempCount(Q) - 1

            NuPos = NuPos + TempCount(Q)

            If TempCount(Q) > 1 Then GoSub QuickSort

          End If

        Next

      End If

    Next

' The array is sorted so let get the last characters and store them

' in the output stream

    ReDim OutStream(FileLength + 2)

    For i = 0 To FileLength

      If F(i) = 1 Then Prefix = i

      If F(i) = 0 Then

        OutStream(i) = ByteArray(FileLength)

      Else

        OutStream(i) = ByteArray(F(i) - 1)

      End If

    Next

    OutStream(FileLength + 1) = Int(Prefix / &H100) And &HFF

    OutStream(FileLength + 2) = Prefix And &HFF

end_Test:

    ReDim ByteArray(UBound(OutStream))

    Call CopyMem(ByteArray(0), OutStream(0), UBound(OutStream) + 1)

    Exit Sub


QuickSort:

    K = 1

    P(K) = L

    W(K) = r

    d = 1

    Do

toploop:

      If r - L < 10 Then GoTo bubsort

      i = L

      J = r

      While J > i

        Y = F(i) + 1: If Y > FileLength Then Y = Y - FileLength - 1

        Z = F(J) + 1: If Z > FileLength Then Z = Z - FileLength - 1

        Do While ByteArray(Y) = ByteArray(Z)

          Y = Y + 1: If Y > FileLength Then Y = Y - FileLength - 1

          Z = Z + 1: If Z > FileLength Then Z = Z - FileLength - 1

        Loop
```

```
        If ByteArray(Y) > ByteArray(Z)
Then

          t = F(J)

          F(J) = F(i)

          F(i) = t

          d = -d

        End If

        If d = -1 Then

          J = J - 1

        Else

          i = i + 1

        End If

      Wend

      J = J + 1

      K = K + 1

      If i - L < r - J Then

        P(K) = J

        W(K) = r

        r = i

      Else

        P(K) = L

        W(K) = i

        L = J

      End If

      d = -d

      GoTo toploop

bubsort:

    If r - L > 0 Then

      For i = L To r

        b = i

        For J = b + 1 To r

          Y = F(J) + 1: If Y > FileLength
Then Y = Y - FileLength - 1

          Z = F(b) + 1: If Z >
FileLength Then Z = Z - FileLength - 1

          Do While ByteArray(Y) =
ByteArray(Z)

            Y = Y + 1: If Y > F ileLength
Then Y = Y - FileLength - 1

            Z = Z + 1: If Z > FileLength
Then Z = Z - FileLength - 1

          Loop

          If ByteArray(Y) <
ByteArray(Z) Then b = J

        Next J

        If i <> b Then

          t = F(b)

          F(b) = F(i)

          F(i) = t

        End If

      Next i

    End If

    L = P(K)

    r = W(K)

    K = K - 1

    Loop Until K = 0

    Return


End Sub
```

```
'Here whe gone restore the BWT-
coded string

Public Sub
BWT_DeCodecArray4(ByteArray()
As Byte)

  Dim TV() As Long

  Dim Spos(255) As Long

  Dim FileLength As Long

  Dim OffSet As Long

  Dim X As Long

  Dim Y As Long

  Dim NuPos As Long

  Dim CharCount(255) As Long

  Dim OutStream() As Byte

  FileLength = UBound(ByteArray)

'read the offset and restore the
original size

  OffSet =
CLng(ByteArray(FileLength - 1)) *
256 + ByteArray(FileLength)
```

```
   ReDim Preserve ByteArray(FileLength - 2)

   FileLength = UBound(ByteArray)

   ReDim OutStream(FileLength)

   ReDim TV(FileLength)

'Lets use the speedsort method to sort the array

'(no need to do it lexicographical)

   For X = 0 To FileLength

      CharCount(ByteArray(X)) = CharCount(ByteArray(X)) + 1

   Next

   NuPos = 0

' Place the items in the sorted arr ay.

   For X = 0 To 255

      Spos(X) = NuPos

      NuPos = NuPos + CharCount(X)

   Next

'Now whe have the original and the sorted array so whe can construct

'a transformation tabel

   For X = 0 To FileLength

      TV(Spos(ByteArray(X))) = X

      Spos(ByteArray(X)) = Spos(ByteArray(X)) + 1

   Next

'with use of the transformation tabel and the offset whe can reconstruct

'the original data

   For X = 0 To FileLength

      OutStream(X) = ByteArray(OffSet)

      OffSet = TV(OffSet)

   Next

   Call CopyMem(ByteArray(0), OutStream(0), UBound(OutStream) + 1)

End Sub
```

## Move To Front

```
Option Explicit

'this is a Move To Front Coder which returns a lot of

'small numbers because when a value is found it will be

'placed at the start of the dictionary

'There are two methods in this module

'

'The first one uses a standard dictionary excisting of all the

'ascii characters

'The second one creates a dictionary while it is coding

'this dictionary has to be stored to get the decoder work



Public Sub MTF_CoderArray(bytes() As Byte, Optional Dictionary As String = "")

   Dim DictString As String

   Dim NewPos As Integer

   Dim X As Long

   If Dictionary = "" Then

      For X = 0 To 255

         DictString = DictString & Chr(X)

      Next

   Else

      DictString = Dictionary

   End If

   For X = 0 To UBound(bytes)

      NewPos = InStr(DictString, Chr(bytes(X)))

      DictString = Chr(bytes(X)) & Left(DictString, NewPos - 1) & Mid(DictString, NewPos + 1)

      bytes(X) = NewPos - 1

   Next

End Sub

PublicSub MTF_DeCoderArray(bytes() As Byte, Optional Dictionary As String = "")

   Dim DictString As String

   Dim NewString As String

   Dim NewPos As Integer
```

```
    Dim X As Long

    If Dictionary = "" Then

        For X = 0 To 255

            DictString = DictString & Chr(X)

        Next

    Else

        DictString = Dictionary

    End If

    For X = 0 To UBound(bytes)

        NewPos = bytes(X) + 1

        bytes(X) = ASC(Mid(DictString, NewPos, 1))

        DictString = Mid(DictString, NewPos, 1) & Left(DictString, NewPos - 1) & Mid(DictString, NewPos + 1)

    Next

End Sub
```

## Global module

```
Public Sub Copy_Orig2Work()

    ReDim WorkArray(UBound(OriginalArray))

    Call CopyMem(WorkArray(0), OriginalArray(0), UBound(OriginalArray) + 1)

End Sub


'copy the workarray to the original array so that whe can apply a

'second compress/coder on the file

Public Sub Copy_Work2Orig()

    ReDim OriginalArray(UBound(WorkArray))

    Call CopyMem(OriginalArray(0), WorkArray(0), UBound(WorkArray) + 1)

End Sub


'This sub is used to start a coder

'whichcoder is the constant of the codertype

'Decode is used to say if we want to code or decode

Public Sub Start_Coder(WichCoder)

    Dim Decode As Boolean

    Dim StTime As Double

    Dim Text As String

    Dim LastUsed As Integer

    If UBound(OriginalArray) = 0 Then

        MsgBox "There is nothing to code/Decode"

        Exit Sub

    End If

    If AutoDecodeIsOn = False Then

        Decode = True

        frmCodeDecode.Show vbModal

        DoEvents

        If CompDecomp = 0 Then Exit Sub

        If CompDecomp = 1 Then Decode = False

    Else

        Decode = True

    End If

    If Decode = True Then

        LastUsed = UsedCodecs(UBound(UsedCodecs))

        If JustLoaded = True Then LastUsed = 0

        If (WichCoder Or 128) <> LastUsed Then

            Text = "This is not coded with the " & CodName(WichCoder) & Chr(13)

            If LastUsed = 0 Then

                Text = Text & "Its not coded at all"

            Else

                If LastUsed > 128 Then

                    Text = Text & "Its coded with the " & CodName(LastUsed And 127)

                Else

                    Text = Text & "Its compressed with " & CompName(LastUsed)

                End If
```

```
        End If
        MsgBox Text
        Exit Sub
    End If
  Else
    LastCoder = WichCoder Or 128
  End If
  LastDeCoded = Decode
  Call Copy_Orig2Work
  If JustLoaded = True Then
    JustLoaded = False
    ReDim UsedCodecs(0)
  End If
  StTime = Timer
  master.MousePointer = MousePointerConstants.vbHourglass
  Select Case WichCoder
    Case 2
      If Decode = False Then Call FrequentShifter_Coder(WorkArray)
      If Decode = True Then Call FrequentShifter_DeCoder(WorkArray)
    Case 3
      If Decode = False Then Call BWT_CodecArray4(WorkArray)
      If Decode = True Then Call BWT_DeCodecArray4(WorkArray)
    Case 6
      If Decode = False Then Call MTF_CoderArray(WorkArray)
      If Decode = True Then Call MTF_DeCoderArray(WorkArray)
  End Select
  master.MousePointer = MousePointerConstants.vbDefault
  If AutoDecodeIsOn = False Then
    Call Show_Statistics(False, WorkArray, Timer - StTime)
  End If
End Sub

'This sub is used to start a compressor
'whichcoder is the constant of the compressortype
'Decode is used to say if we want to compress or decompress
Public Sub Start_Compressor(WichCompressor)
  Dim Decompress As Boolean
  Dim Dummy As Boolean
  Dim StTime As Double
  Dim Text As String
  Dim LastUsed As Integer
  If UBound(OriginalArray) = 0 Then
    MsgBox "There is nothing to compress/Decompress"
    Exit Sub
  End If
  If AutoDecodeIsOn = False Then
    Decompress = True
    frmCompDecomp.Show vbModal
    DoEvents
    If CompDecomp = 0 Then Exit Sub
    If CompDecomp = 1 Then Decompress = False
  Else
    Decompress = True
  End If
  If Decompress = True Then
    LastUsed = UsedCodecs(UBound(UsedCodecs))
    If JustLoaded = True Then LastUsed = 0
    If WichCompressor <> LastUsed Then
      Text = "This is not compressed with " & CompName(WichCompressor) & "." & Chr(13)
      If LastUsed = 0 Then
        Text = Text & "Its not compressed at all"
```

```
        Else
          If LastUsed > 128 Then
            Text = Text & "Its coded with the " & CodName(LastUsed And 127)
          Else
            Text = Text & "Its compressed with " & CompName(LastUsed)
          End If
        End If
        MsgBox Text
        Exit Sub
      End If
    Else
      LastCoder = WichCompressor
    End If
    Call Copy_Orig2Work
    LastDeCoded = Decompress
    If JustLoaded = True Then
      JustLoaded = False
        ReDim UsedCodecs(0)
    End If
    master.MousePointer = MousePointerConstants.vbHourglass
    StTime = Timer
    Select Case WichCompressor
      Case 38
        If Decompress = False Then Call Compress_LBE(WorkArray, 1)
        If Decompress = True Then Call DeCompress_LBE(WorkArray, 1)
    End Select
    master.MousePointer = MousePointerConstants.vbDefault
    If AutoDecodeIsOn = False Then
      Call Show_Statistics(False, WorkArray, Timer - StTime)
    End If
End Sub

'this sub is used to load a chosen file

Public Sub load_File(Name As String)
    Dim FreeNum As Integer
    If Name = "" Then Exit Sub
    FreeNum = FreeFile
    Open Name For Binary As #FreeNum
    ReDim OriginalArray(0 To LOF(FreeNum) - 1)
    Get #FreeNum, , OriginalArray()
    Close #FreeNum
    JustLoaded = True
    Call Split_Header_From_File(OriginalArray)
    master.Caption = "Test Programm For Compressors [file = " & LoadFileName & "]"
    OriginalSize = UBound(OriginalArray) + 1
    Call Show_Statistics(True, OriginalArray)
End Sub

'this sub is used to see if the file just loaded is a file which is

'stored by this programm and is already coded/compressed

Private Sub Split_Header_From_File(ByteArray() As Byte)
    Dim HeadText As String
    Dim X As Integer
    Dim CodecsUsed As Integer
    Dim Version As String
    Dim InPos As Long
    If UBound(ByteArray) < 3 Then Exit Sub  'original file to small
    InPos = UBound(ByteArray)
    For X = 0 To 2
      HeadText = HeadText & Chr(ByteArray(InPos))
      InPos = InPos - 1
    Next
    If HeadText <> "UCF" Then Exit Sub  'this is an un-UCF'ed file
```

```
    Version = Chr(ByteArray(InPos))
    InPos = InPos - 1
    Select Case Version
      Case "0"
        CodecsUsed = ByteArray(InPos)
        InPos = InPos - 1
        ReDim UsedCodecs(CodecsUsed)
        For X = 1 To CodecsUsed
          UsedCodecs(X) = ByteArray(InPos)
          InPos = InPos - 1
        Next
        ReDim Preserve ByteArray(InPos)
    End Select
    ReDim WorkArray(0)
    For X = 0 To 255
      master.Bars(1 * 256 + X).Visible = False
    Next
    master.AscTab(1).Clear
    master.FreqTab(1).Clear
    master.FileSize(1).Caption = " "
    master.MaxValue(1).Caption = "Maximum"
    master.MidValue(1).Caption = "Medium"
    master.LowValue(1).Caption = "Lowest"
    JustLoaded = False
End Sub


'this sub is used to save a file

'if the file was coded/compressed the types of coders/compressors used

'will be saved with the file so that we can recall it later

Public Sub Save_File_As(ByteArray() As Byte, source As Boolean)
    Dim FileNr As Integer
    Dim HeadArray() As Byte
    Dim OutHead As Integer
    Dim HeadText As String
    Dim Answer As Integer
    Dim CodecsUsed As Integer
    Dim SaveName As String
    Dim ExtPos As Integer
    Dim Temp As Integer
    Dim X As Integer
    If UBound(ByteArray) = 0 Then
      MsgBox "There is nothing to be saved"
      Exit Sub
    End If
    If source = False And LastCoder <> 0 Then Call AddCoder2List(LastCoder)
    If UBound(UsedCodecs) = 0 And UBound(ByteArray) = UBound(OriginalArray) Then
      Answer = MsgBox("The file to save is the same as the original file" & Chr(13) & "Still want to save this file", vbYesNo + vbExclamation)
      If Answer = vbNo Then
        Exit Sub
      End If
    End If
Ask_SaveName:
    SaveName = ""
    master.Cdlg.DialogTitle = "Type in the name you want to save with"
    master.Cdlg.FileName = ""
    master.Cdlg.ShowSave
    SaveName = master.Cdlg.FileName
    If SaveName = "" Then
      If source = False And LastCoder <> 0 Then
        ReDim Preserve UsedCodecs(UBound(UsedCodecs) - 1)
        LastCoder = UsedCodecs(UBound(UsedCodecs))
```

```
    End If

    Exit Sub

  End If

  Temp = 0

  Do

    ExtPos = Temp

    Temp = InStr(ExtPos + 1, SaveName, ".")

  Loop While Temp <> 0

  If ExtPos = 0 Or ExtPos < Len(SaveName) - 5 Then

    SaveName = SaveName & ".txt"

  End If

'store the header in reversed order at the end of the file

  HeadText = "UCF0"

  If LastCoder = 0 And source = False Then

    CodecsUsed = 0

  Else

    CodecsUsed = UBound(UsedCodecs)

  End If

  ReDim HeadArray(4 + CodecsUsed)

  OutHead = 0

  For X = CodecsUsed To 1 Step -1

    HeadArray(OutHead) = UsedCodecs(X)

    OutHead = OutHead + 1

  Next

  HeadArray(OutHead) = CodecsUsed

  OutHead = OutHead + 1

  For X = Len(HeadText) To 1 Step -1

    HeadArray(OutHead) = ASC(Mid(HeadText, X, 1))

    OutHead = OutHead + 1

  Next

  FileNr = FreeFile

  If Dir(SaveName, vbNormal) <> "" Then

    Answer = MsgBox("File already exists" & Chr(13) & Chr(13) & "Overwrite", vbCritical + vbYesNo)

    If Answer = vbNo Then

      GoTo Ask_SaveName

    End If

    Kill SaveName 'first remove it otherwise size is not adjusted

  End If

  Open SaveName For Binary As #FileNr

  Put #FileNr, , ByteArray()

  If CodecsUsed > 0 Then

    Put #FileNr, , HeadArray()

  End If

  Close #FileNr

End Sub


'this sub is used to show the statistics of a file
'it can display both the original as the workarray

Public Sub Show_Statistics(OrgData As Boolean, Data() As Byte, Optional TimeUsed As Double = 0)

  Dim StatWindow As Integer

  Dim Frequentie(255) As Long

  Dim SortFreq(1, 255) As Long

  Dim Counts() As Long

  Dim X As Long

  Dim Minval As Long

  Dim Maxval As Long

  Dim next_offset As Long

  Dim this_count As Long

  Dim HeightValue As Double

  Dim Entry As String

  Dim NewSize As String

  Dim NuSize As Long

  Dim BPB As String

  If OrgData = False Then StatWindow = 1
```

```
   NuSize = UBound(Data) + 1

   BPB = Format(((NuSize * 8) / OriginalSize), "###0.000") & " bpb"

   NewSize = NuSize & " Bytes [ " & Format(100 - (OriginalSize - NuSize) / OriginalSize * 100, "##0.00") & "% ] "

   If TimeUsed > 0 Then

      NewSize = NewSize & BPB & "  " & Format(TimeUsed, "###0.00") & " Sec."

   End If

   For X = 0 To UBound(Data)

      Frequentie(Data(X)) = Frequentie(Data(X)) + 1

   Next

   Minval = UBound(Data)

   For X = 0 To 255

      If Minval > Frequentie(X) Then Minval = Frequentie(X)

      If Maxval < Frequentie(X) Then Maxval = Frequentie(X)

   Next

' Lets use the counting sort to sort them into another array

' Create the Counts array.

   ReDim Counts(Minval To Maxval)

' Count the items.

   For X = 0 To 255

      Counts(Frequentie(X)) = Counts(Frequentie(X)) + 1

   Next X

' Convert the counts into offsets.

   next_offset = 0

   For X = Maxval To Minval Step -1

      this_count = Counts(X)

      Counts(X) = next_offset

      next_offset = next_offset + this_count

   Next X

' Place the items in the sorted array.

   For X = 0 To 255

      SortFreq(0, Counts(Frequentie(X))) = Frequentie(X)

      SortFreq(1, Counts(Frequentie(X))) = X

      Counts(Frequentie(X)) = Counts(Frequentie(X)) + 1

   Next X

'Create the graphics view

   HeightValue = (SortFreq(0, 0) - SortFreq(0, 255)) / master.Graphic(StatWindow).Height

   For X = 0 To 255

      If Frequentie(X) - SortFreq(0, 255) <> 0 Then

         master.Bars(StatWindow * 256 + X).Visible = True

         master.Bars(StatWindow * 256 + X).Y1 = master.Bars(StatWindow * 256 + X).Y2 - (Frequentie(X) - SortFreq(0, 255)) / HeightValue

         master.Bars(StatWindow * 256 + X).BorderColor = RGBColor(X Mod 7)

      Else

         master.Bars(StatWindow * 256 + X).Visible = False

      End If

   Next


master.MaxValue(StatWindow).Caption = SortFreq(0, 0)


master.MidValue(StatWindow).Caption = Int((SortFreq(0, 0) - SortFreq(0, 255)) / 2) + SortFreq(0, 255)

   master.LowValue(StatWindow) = SortFreq(0, 255)


master.FileSize(StatWindow).Caption = NewSize

'Create the statistics view

   master.AscTab(StatWindow).Clear


master.FreqTab(StatWindow).Clear

   Entry = "Index    Freq."


master.AscTab(StatWindow).AddItem Entry
```

```
   For X = 0 To 255

      Entry = Format(X, "##0") & Chr(9) & Frequentie(X)


master.AscTab(StatWindow).AddItem Entry

   Next

   Entry = "Index" & Chr(9) & "Ascii" & Chr(9) & "Freq."


master.FreqTab(StatWindow).AddItem Entry

   For X = 0 To 255

      Entry = Format(X, "##0") & Chr(9) & SortFreq(1, X) & Chr(9) & SortFreq(0, X)


master.FreqTab(StatWindow).AddItem Entry

   Next

End Sub



'this sub is used to show the contents of the file

'its used for both the original as the workarray

Public Sub Show_Contents(ByteArray() As Byte)

   Dim X As Long

   Dim Y As Integer

   Dim AddData As String

   Dim AddText As String

   Dim Data As Byte

   Dim Text As String

   On Error GoTo No_Data

   X = UBound(ByteArray)

   If X = 0 Then

      MsgBox "Ther is nothing to see because there is no data"

      Exit Sub

   End If

   On Error GoTo 0

   frmViewContents.Show


frmViewContents.lstContents.Clear

   For X = 0 To UBound(ByteArray) Step 16

      AddData = String(61, " ")

      Mid(AddData, 35, 1) = "|"

      AddText = String(16, " ")

      Mid(AddData, 1, 9) = Right("00000000" & Hex(X), 8) & ":"

      For Y = 0 To 15

         If X + Y <= UBound(ByteArray) Then

            Data = ByteArray(X + Y)

            Mid(AddData, 12 + (3 * Y), 2) = Right("0" & Hex(Data), 2)

            If Data < 28 Then Text = Chr(1) Else Text = Chr(Data)

            Mid(AddText, Y + 1, 1) = Text

         End If

      Next

      If frmViewContents.Visible = True Then

frmViewContents.lstContents.AddItem AddData & AddText

      Else

         Exit Sub

      End If

      If X Mod 500 * 16 = 0 Then DoEvents

   Next

   DoEvents

No_Data:

End Sub



'this sub is used to store a coder/compressor type into an array

'so that we can keep up which coders/compressors are used to get to

'the last file whe have standing in the original array

Public Sub AddCoder2List(CodeNumber As Integer)
```

```
    JustLoaded = False

    If LastDeCoded = True Then

      If UBound(UsedCodecs) > 0 Then

        ReDim Preserve UsedCodecs(UBound(UsedCodecs) - 1)

        LastCoder = UsedCodecs(UBound(UsedCodecs))

      End If

      Exit Sub

    End If

    ReDim Preserve UsedCodecs(UBound(UsedCodecs) + 1)


UsedCodecs(UBound(UsedCodecs)) = CodeNumber

End Sub



'this sub is used to decode/uncompress automaticly without the user
'having search which type of coder/compressor was used

Public Sub Auto_Decode_Depack()

    Dim X As Integer

    Dim CodeNumber As Integer

    If UBound(OriginalArray) = 0 Then

      MsgBox "There is nothing to Decode/Decompress"

      Exit Sub

    End If

    If UBound(UsedCodecs) = 0 Or JustLoaded = True Then

      MsgBox "This file was'nt Coded/Compressed"

      Exit Sub

    End If

    CodeNumber = UsedCodecs(UBound(UsedCodecs))

    AutoDecodeIsOn = True

    If CodeNumber > 128 Then

      Call Start_Coder(CodeNumber And 127)

    Else

      Call Start_Compressor(CodeNumber)

    End If

    Call Copy_Work2Orig

    ReDim WorkArray(0)

    For X = 0 To 255

      master.Bars(1 * 256 + X).Visible = False

    Next

    master.AscTab(1).Clear

    master.FreqTab(1).Clear

    master.FileSize(1).Caption = " "

    master.MaxValue(1).Caption = "Maximum"

    master.MidValue(1).Caption = "Medium"

    master.LowValue(1).Caption = "Lowest"

    AutoDecodeIsOn = False

    If UBound(UsedCodecs) > 0 Then

      ReDim Preserve UsedCodecs(UBound(UsedCodecs) - 1)

      LastCoder = UsedCodecs(UBound(UsedCodecs))

    End If

    Call Show_Statistics(True, OriginalArray)

End Sub



'this sub is used to compare the original array with the workarray

Public Sub
```

**Private Sub CopyWorkToOrg_Click()**

```
    If UBound(WorkArray) = 0 Then

      MsgBox "There is nothing to copy"

      Exit Sub

    End If

    Call Copy_Work2Orig

    Call AddCoder2List(LastCoder)
```

```
   Call Show_Statistics(True, OriginalArray)

End Sub


Private Sub ExitProg_Click()

   End

End Sub


Private Sub Form_Load()

   Dim X As Integer

   Dim Y As Integer

   Dim MaxWidth As Double

   For X = 0 To 1

      MaxWidth = Graphic(X).Width / 256

      For Y = 0 To 255

         Bars(X * 256 + Y).BorderWidth = 1

         Bars(X * 256 + Y).BorderStyle = 1

         Bars(X * 256 + Y).X1 = Graphic(X).Left + Y * MaxWidth

         Bars(X * 256 + Y).X2 = Bars(X * 256 + Y).X1

         Bars(X * 256 + Y).Y2 = Graphic(X).Top + Graphic(X).Height

         Bars(X * 256 + Y).Y1 = Bars(X * 256 + Y).Y2

         Bars(X * 256 + Y).Visible = True

      Next

   Next

   RGBColor(0) = vbBlue

   RGBColor(1) = vbCyan

   RGBColor(2) = vbGreen

   RGBColor(3) = vbMagenta

   RGBColor(4) = vbRed

   RGBColor(5) = vbYellow

   RGBColor(6) = vbWhite

   Call Init_CoderNameDataBase

   ReDim OriginalArray(0)

   ReDim WorkArray(0)

   ReDim UsedCodecs(0)

End Sub


Private Sub Form_Unload(Cancel As Integer)

   End

End Sub


Private Sub FreqShift_Click()

   Call Start_Coder(Coder_FrequentieShifter)

End Sub


Private Sub LBE_Click()

   Call Start_Compressor(Compressor_LBE_Flat)

End Sub
```

**Private Sub Load_Click()**

```
   Dim OldFileName As String

   OldFileName = LoadFileName

   Cdlg.DialogTitle = "Select the file you want to explore"

   Cdlg.FileName = ""

   Cdlg.ShowOpen

   LoadFileName = Cdlg.FileName

   Call load_File(LoadFileName)

   If LoadFileName = "" Then LoadFileName = OldFileName

End Sub


Private Sub MTF_Click()

   Call Start_Coder(Coder_MTF_No_Header)

End Sub
```

**Private Sub RestoreOrig_Click()**

```
   If LoadFileName = "" Then
```

```
    MsgBox "There was'nt yet original data"
    Exit Sub
  End If
  Call load_File(LoadFileName)
End Sub
```

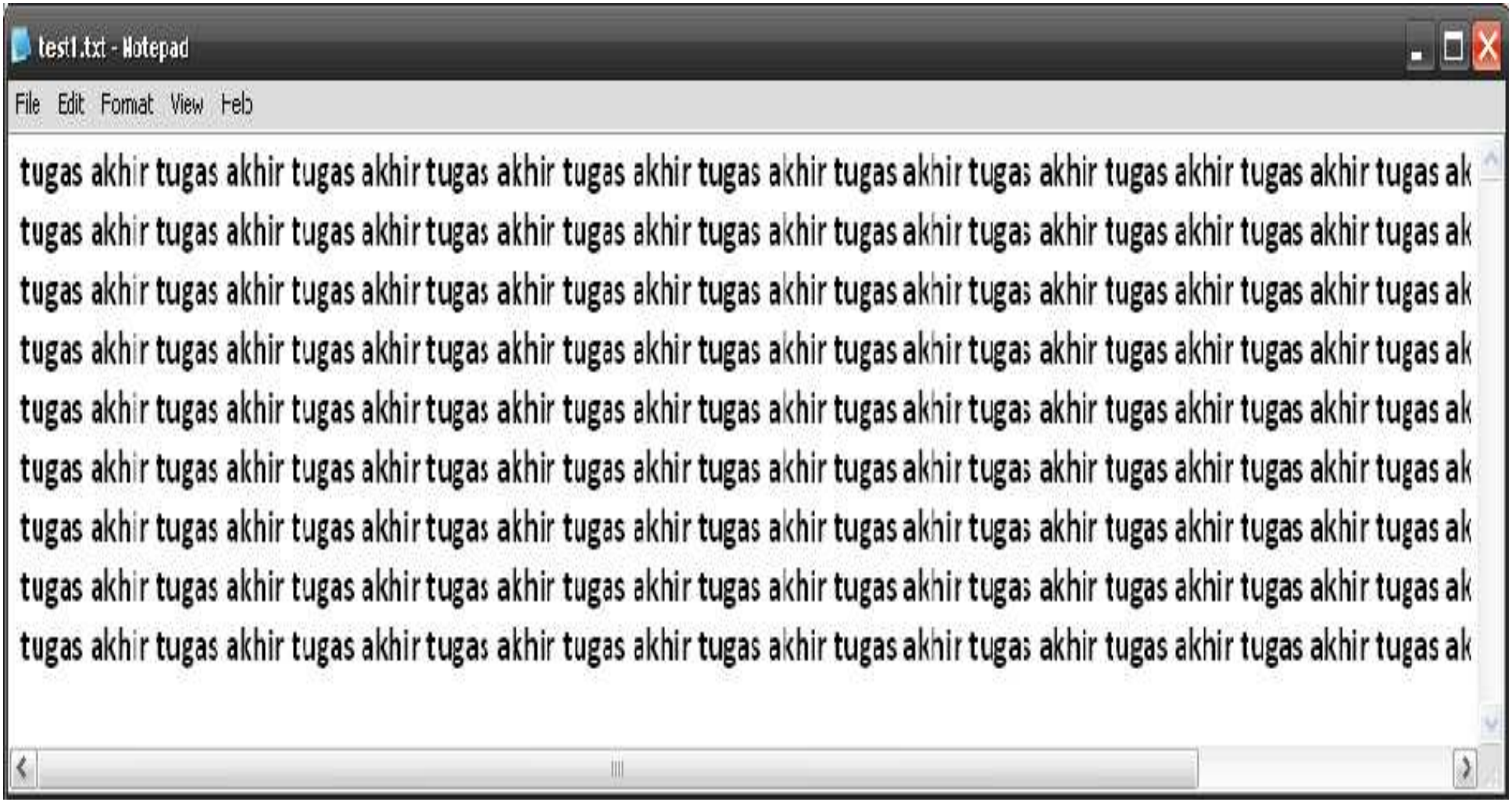
**Private Sub Save_Click()**

```
  Call Save_File_As(WorkArray, False)
End Sub
```

# LAMPIRAN B

# ISI FILE UNTUK PERCOBAAN

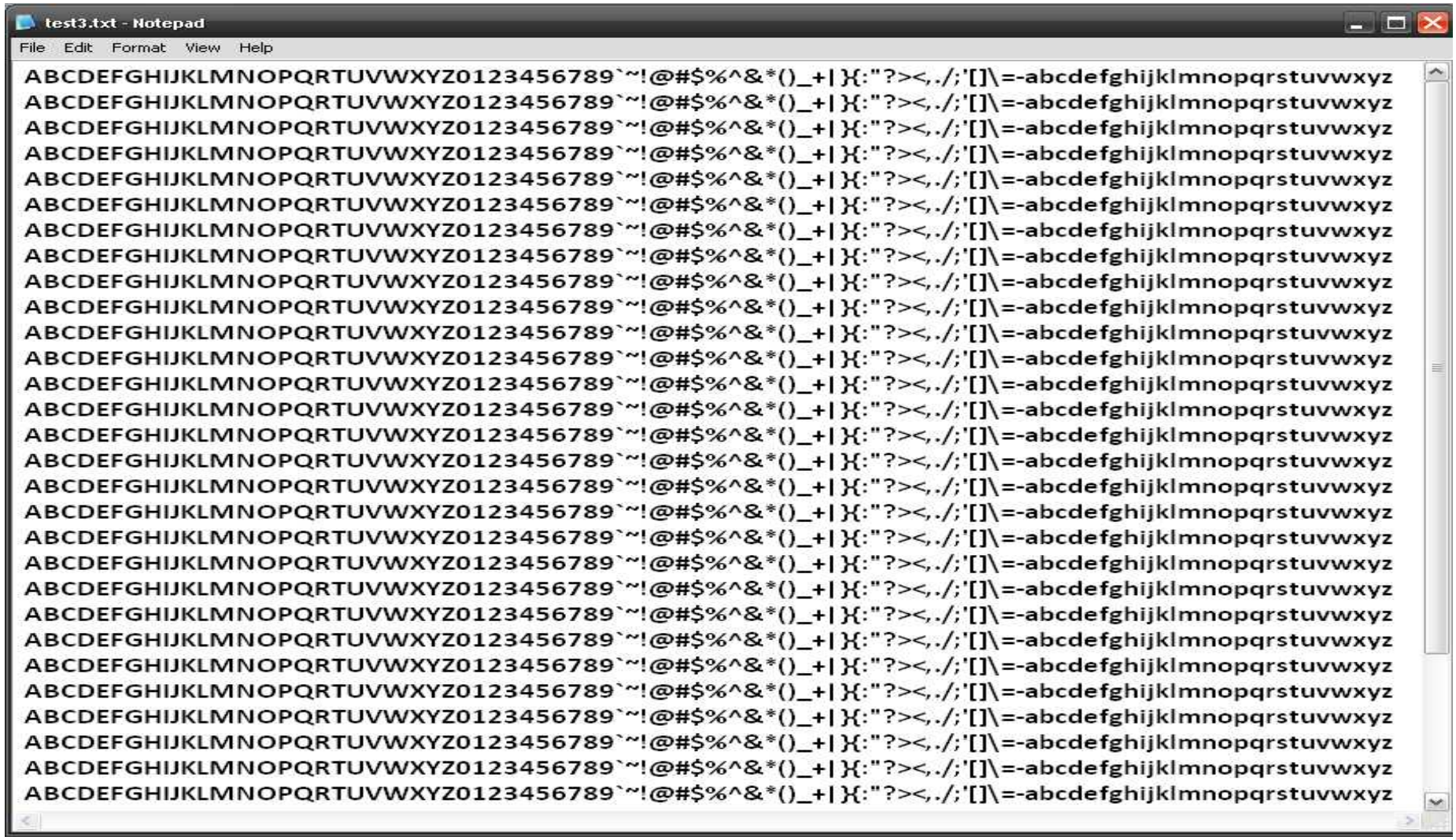
1. Test 1


test1.txt - Notepad
File Edit Format View Help
tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas ak
tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas ak
tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas ak
tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas ak
tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas ak
tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas ak
tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas ak
tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas ak
tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas akhir tugas ak

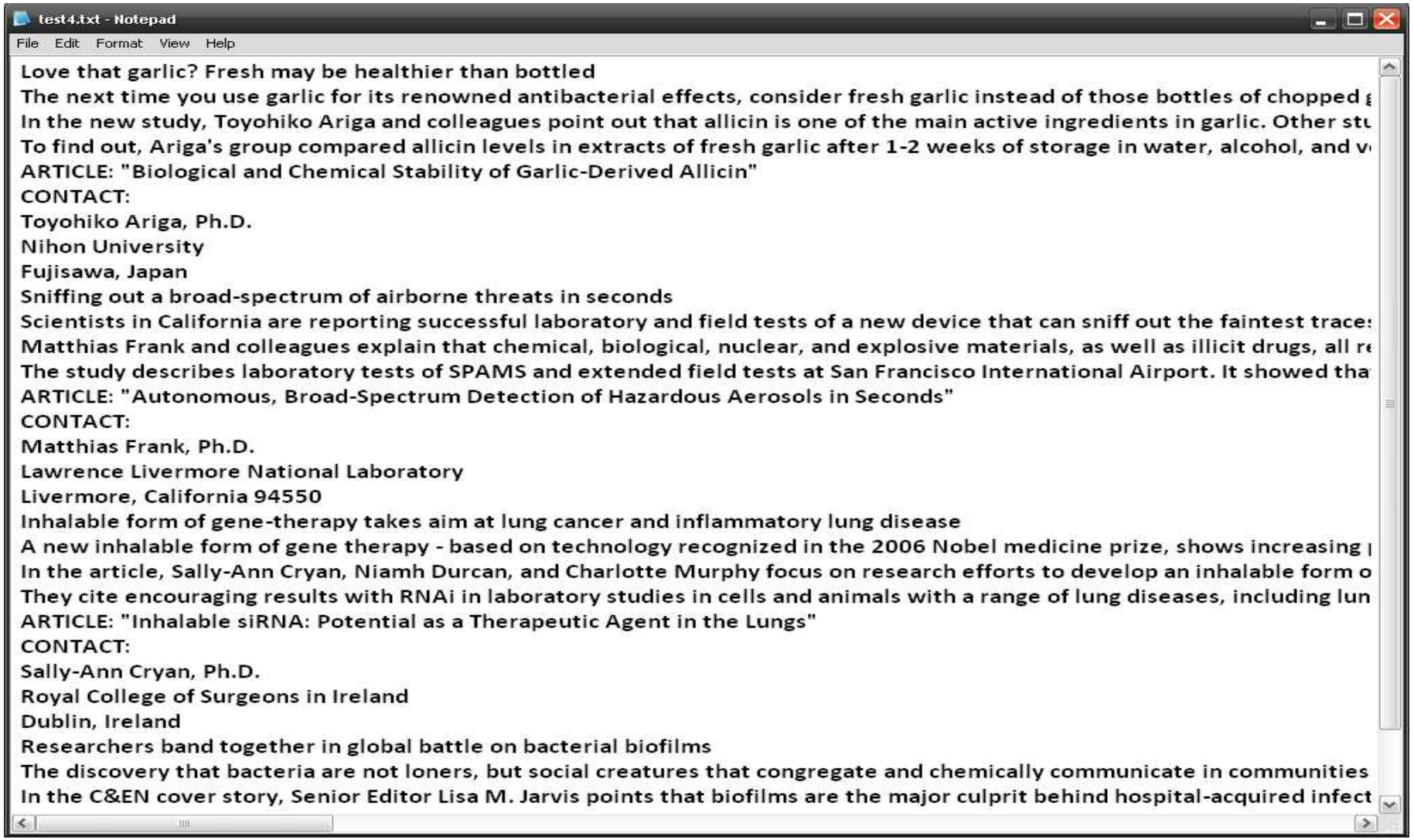
## 2. Test 2


test2.txt - Notepad
File Edit Format View Help
aaaaaaaaaaaaaaaaaaa bbbbbbbbbbbbbbbbbbb ccccccccccccccccccc
ddddddddddddddddddddd eeeeeeeeeeeeeeeeeee fffffffffffffffffff
ggggggggggggggggggg hhhhhhhhhhhhhhhhhhh iiiiiiiiiiiiiiiiii
jjjjjjjjjjjjjjjjj kkkkkkkkkkkkkkkkkkk lllllllllllllllllll mmmmmmmmmmmmmmmmmmmm
nnnnnnnnnnnnnnnnnnn ooooooooooooooooooo
ppppppppppppppppppp qqqqqqqqqqqqqqqqqqq rrrrrrrrrrrrrrrrrr
ssssssssssssssssss tttttttttttttttttt uuuuuuuuuuuuuuuuuuu vvvvvvvvvvvvvvvvvv
wwwwwwwwwwwwwwwwwwww xxxxxxxxxxxxxxxxxxx yyyyyyyyyyyyyyyyyyy zzzzzzzzzzzzzzzzzzz
11111111111111111111 22222222222222222222 33333333333333333333 44444444444444444444
55555555555555555555 66666666666666666666 77777777777777777777
88888888888888888888 99999999999999999999 00000000000000000000
~~~~~~~~~~~~~~~~~~~~ !!!!!!!!!!!!!!!!!! @@@@@@@@@@@@@@@@@@@@@@
#################### $$$$$$$$$$$$$$$$$$$$ %%%%%%%%%%%%%%%%%%%%
^^^^^^^^^^^^^^^^^^^^ &&&&&&&&&&&&&&&&&&&& ********** (((((((((((((((((( ))))))))))))))))))
_____ +++++++++++++++++++ ========== ---------- ~~~~~~~~~~~~~~~~~~~~ [[[[[[[[[[[[[[[[[[
]]]]]]]]]]]]]]]]] {{{{{{{{{{{{{{{{{{{{ }}}}}}}}}}}}}}}}}} \\\\\\\\\\\\\\\\\\\\ |||||||||||||||||| ;;;;;;;;;;;;;;;;;;;; """""""""""""""""""
:::::::::::::::::::: ''''''''''''''''''' ,,,,,,,,,,,,,,,,,,, <<<<<<<<<<<<<<<<<<<< .......... >>>>>>>>>>>>>>>>>>>> ////////////////////
AAAAAAAAAAAAAAAAAAA BBBBBBBBBBBBBBBBBBB CCCCCCCCCCCCCCCCCCC DDDDDDDDDDDDDDDDDDD EEEEEE
FFFFFFFFFFFFFFFFFFF GGGGGGGGGGGGGGGGGGG HHHHHHHHHHHHHHHHHHH IIIIIIIIIIIIIIIIII JJJJJJJJJJJJJJJJJJ KKKKKKK
LLLLLLLLLLLLLLLLLLL MMMMMMMMMMMMMMMMMMMM NNNNNNNNNNNNNNNNNNNN OOOOOOOOOOOOOO
PPPPPPPPPPPPPPPPPPP QQQQQQQQQQQQQQQQQQQQ RRRRRRRRRRRRRRRRRRR SSSSSSSSSSSSSSSSSS TTTTTTTTTT
UUUUUUUUUUUUUUUUUUU VVVVVVVVVVVVVVVVVVV WWWWWWWWWWWWWWWWWWWW
XXXXXXXXXXXXXXXXXXX YYYYYYYYYYYYYYYYYYY ZZZZZZZZZZZZZZZZZZZ
~~~~~~~~~~~~~~~~~~~~

## 3. Test 3


test3.txt - Notepad
File Edit Format View Help
ABCDEFGHIJKLMNOPQRTUVWXYZ0123456789`~!@#$%^&*()_+|}{:"?><,./;'[]\=-abcdefghijklmnopqrstuvwxyz

## 4. Test 4


test4.txt - Notepad
File Edit Format View Help
Love that garlic? Fresh may be healthier than bottled
The next time you use garlic for its renowned antibacterial effects, consider fresh garlic instead of those bottles of chopped
In the new study, Toyohiko Ariga and colleagues point out that allicin is one of the main active ingredients in garlic. Other stu
To find out, Ariga's group compared allicin levels in extracts of fresh garlic after 1-2 weeks of storage in water, alcohol, and v
ARTICLE: "Biological and Chemical Stability of Garlic-Derived Allicin"
CONTACT:
Toyohiko Ariga, Ph.D.
Nihon University
Fujisawa, Japan
Sniffing out a broad-spectrum of airborne threats in seconds
Scientists in California are reporting successful laboratory and field tests of a new device that can sniff out the faintest trace
Matthias Frank and colleagues explain that chemical, biological, nuclear, and explosive materials, as well as illicit drugs, all r
The study describes laboratory tests of SPAMS and extended field tests at San Francisco International Airport. It showed tha
ARTICLE: "Autonomous, Broad-Spectrum Detection of Hazardous Aerosols in Seconds"
CONTACT:
Matthias Frank, Ph.D.
Lawrence Livermore National Laboratory
Livermore, California 94550
Inhalable form of gene-therapy takes aim at lung cancer and inflammatory lung disease
A new inhalable form of gene therapy - based on technology recognized in the 2006 Nobel medicine prize, shows increasing
In the article, Sally-Ann Cryan, Niamh Durcan, and Charlotte Murphy focus on research efforts to develop an inhalable form o
They cite encouraging results with RNAi in laboratory studies in cells and animals with a range of lung diseases, including lun
ARTICLE: "Inhalable siRNA: Potential as a Therapeutic Agent in the Lungs"
CONTACT:
Sally-Ann Cryan, Ph.D.
Royal College of Surgeons in Ireland
Dublin, Ireland
Researchers band together in global battle on bacterial biofilms
The discovery that bacteria are not loners, but social creatures that congregate and chemically communicate in communities
In the C&EN cover story, Senior Editor Lisa M. Jarvis points that biofilms are the major culprit behind hospital-acquired infect

## 5. Test 5

test5.txt - Notepad

File Edit Format View Help

BAB III
PERANCANGAN SOFTWARE

Pada bab ini akan dibahas mengenai perancangan software game "menusuk balon" baik jalannya permainan, algoritma pemrc

3.1 Perencanaan Jalannya Permainan
3.1.1 Posisi Kamera Web
Terdapat beberapa posisi kamera web yang dapat digunakan untuk bermain game "menusuk balon". Namun dianjurkan untuk

Gambar 3.1 Posisi kamera web

3.1.2 Jarak Tongkat Penusuk
Kamera web menangkap gambar berupa citra 2D, sehingga, jika tongkat penusuk didekatkan ke kamera web maka akan didap

3.2. Perencanaan Sistem
Jika dilihat dari diagram blok game "menusuk balon" di halaman 2, terdapat beberapa hal yang dibutuhkan dalam pemrosesar
v Hardware :
ü Personal Computer
ü Kamera Web
v Software :
ü "EZTwain" sebagai Camera Capture Driver
ü Program Visual Basic 6.0
ü program permainan
Penempatan posisi kamera dengan posisi yang sesuai yang dihubungkan dengan komputer yang akan mengolah gambar. Hasil

test5.txt - Notepad

File Edit Format View Help

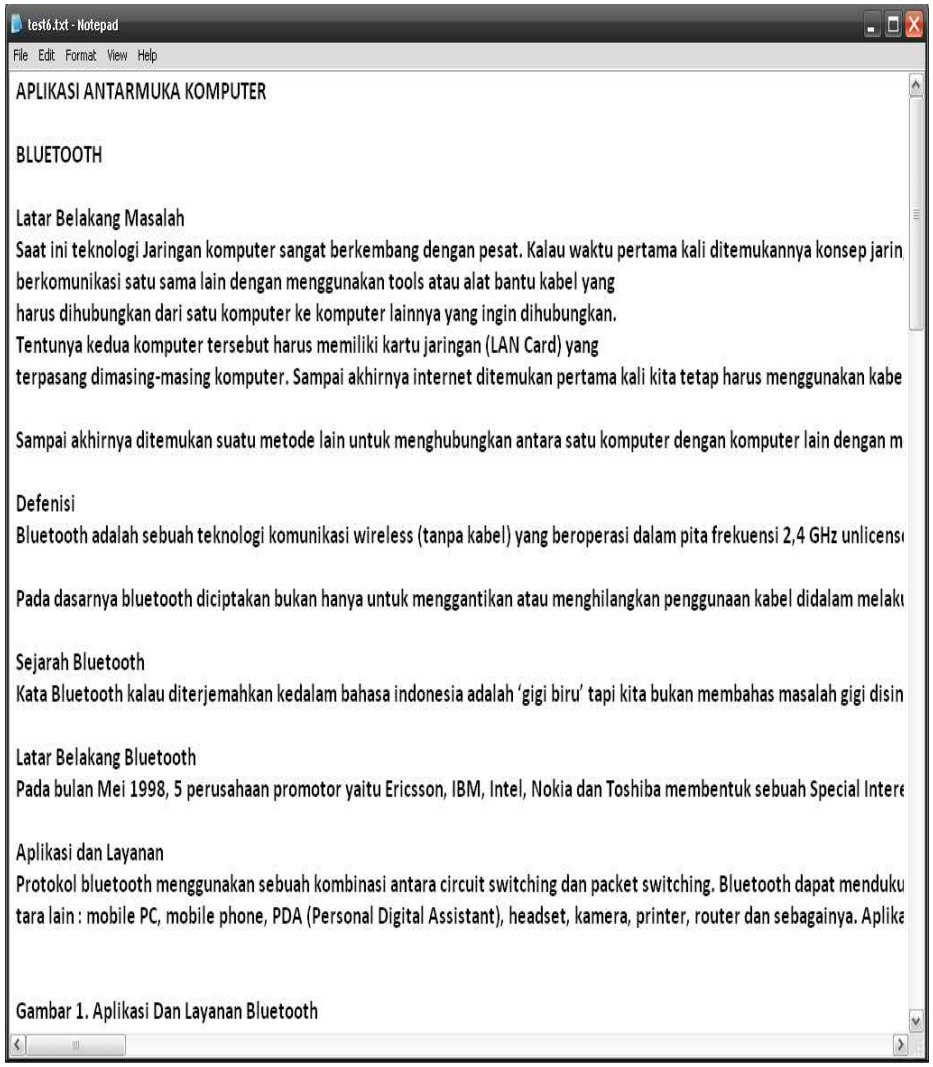
3.3.1.5 Memeriksa irisan antara gambar tangan & gambar balon
Secara hirarki berada pada tingkat ketiga yaitu high –level scene interpretation via images. Ujung tongkat penusuk yang dida
If (i>= Balon.Left) And (i<= Balon.Left + Balon.Width) And (j <= Balon.Top + Height) And (j >= Balon.Top) And (Picture2.Visible

3.3.1.6 Mengakhiri jalannya permainan
Private Sub Command1_Click()
End
End Sub

3.3.2 Variabel yang digunakan dan keterangannya
Daftar variabel-variabel ini dibuat untuk mempermudah memahami program game yang akan dijelaskan pada subbab selanju

i= koordinat x sementara / akhir dari "penusuk balon"
j= koordinat y sementara / akhir dari "penusuk balon"
x= koordinat x dari balon-balon
y= koordinat y dari balon-balon
r= derajat kemerahan dari pixel yang ditunjuk
g= derajat kehijauan dari pixel yang ditunjuk
b= derajat kebiruan dari pixel yang ditunjuk
pix= variabel yang digunakan untuk mengatasi bug
Q= pergerakan vertikal balon berupa counter 1 sampai 8 ( dengan 8 = 1)
z= hasil grayscale dari pixel yang ditunjuk
Picture1= hasil capture kamera web diletakkan disini
Picture2, Picture3, Picture4 = gambar balon dan animasi balon pecah diletakkan disini
balon mini.jpg= gambar balon saat belum pecah, diletakkan di picture 2, 3 dan 4 (gambar 3.3.a )
balon mini2.jpg= salah satu dari serangkaian gambar animasi balon pecah (gambar 3.3.b )
balon mini3.jpg = salah satu dari serangkaian gambar animasi balon pecah (gambar 3.3.c )
balon mini4.jpg = salah satu dari serangkaian gambar animasi balon pecah (gambar 3.3.d )

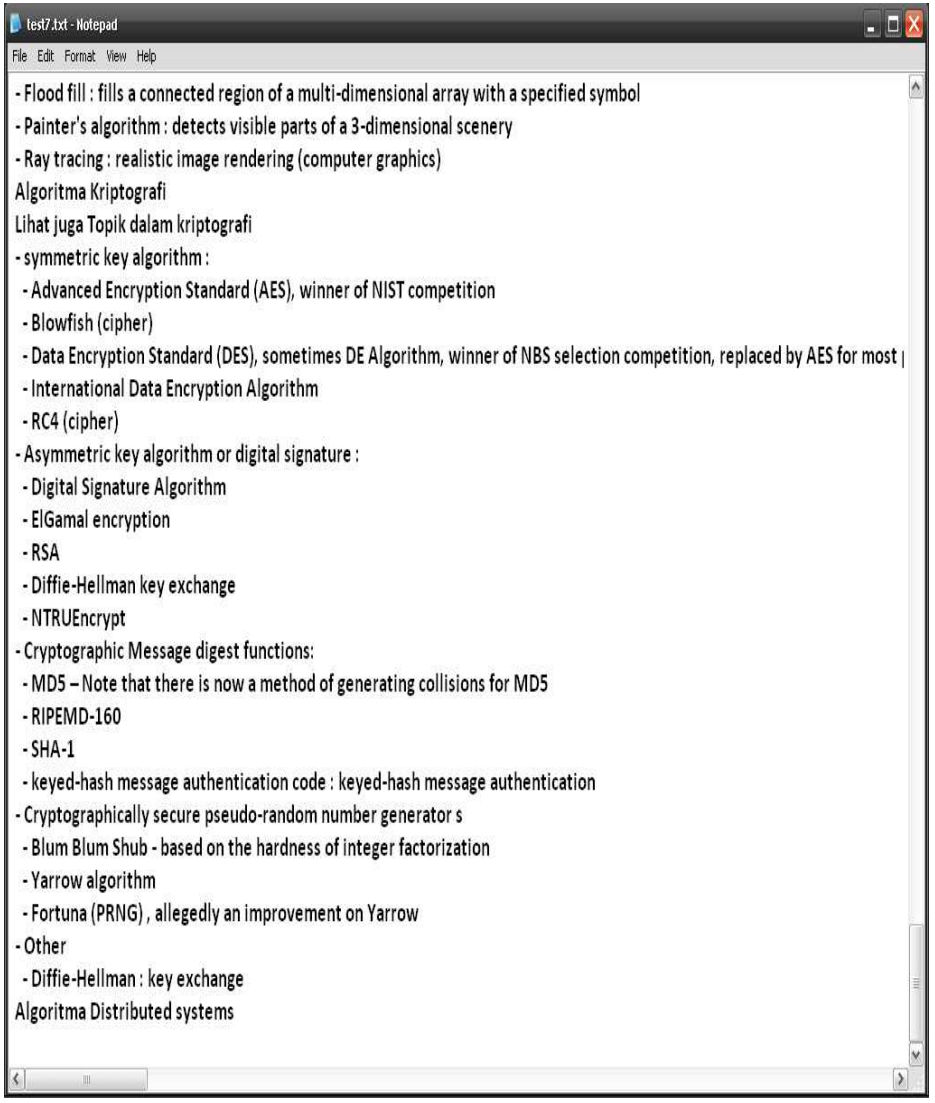
## 6. Test 6


test6.txt - Notepad
File Edit Format View Help
APLIKASI ANTARMUKA KOMPUTER
BLUETOOTH
Latar Belakang Masalah
Saat ini teknologi Jaringan komputer sangat berkembang dengan pesat. Kalau waktu pertama kali ditemukannya konsep jarin
berkomunikasi satu sama lain dengan menggunakan tools atau alat bantu kabel yang
harus dihubungkan dari satu komputer ke komputer lainnya yang ingin dihubungkan.
Tentunya kedua komputer tersebut harus memiliki kartu jaringan (LAN Card) yang
terpasang dimasing-masing komputer. Sampai akhirnya internet ditemukan pertama kali kita tetap harus menggunakan kabe
Sampai akhirnya ditemukan suatu metode lain untuk menghubungkan antara satu komputer dengan komputer lain dengan m
Defenisi
Bluetooth adalah sebuah teknologi komunikasi wireless (tanpa kabel) yang beroperasi dalam pita frekuensi 2,4 GHz unlicens
Pada dasarnya bluetooth diciptakan bukan hanya untuk menggantikan atau menghilangkan penggunaan kabel didalam melak
Sejarah Bluetooth
Kata Bluetooth kalau diterjemahkan kedalam bahasa indonesia adalah 'gigi biru' tapi kita bukan membahas masalah gigi disin
Latar Belakang Bluetooth
Pada bulan Mei 1998, 5 perusahaan promotor yaitu Ericsson, IBM, Intel, Nokia dan Toshiba membentuk sebuah Special Intere
Aplikasi dan Layanan
Protokol bluetooth menggunakan sebuah kombinasi antara circuit switching dan packet switching. Bluetooth dapat menduku
tara lain : mobile PC, mobile phone, PDA (Personal Digital Assistant), headset, kamera, printer, router dan sebagainya. Aplika
Gambar 1. Aplikasi Dan Layanan Bluetooth

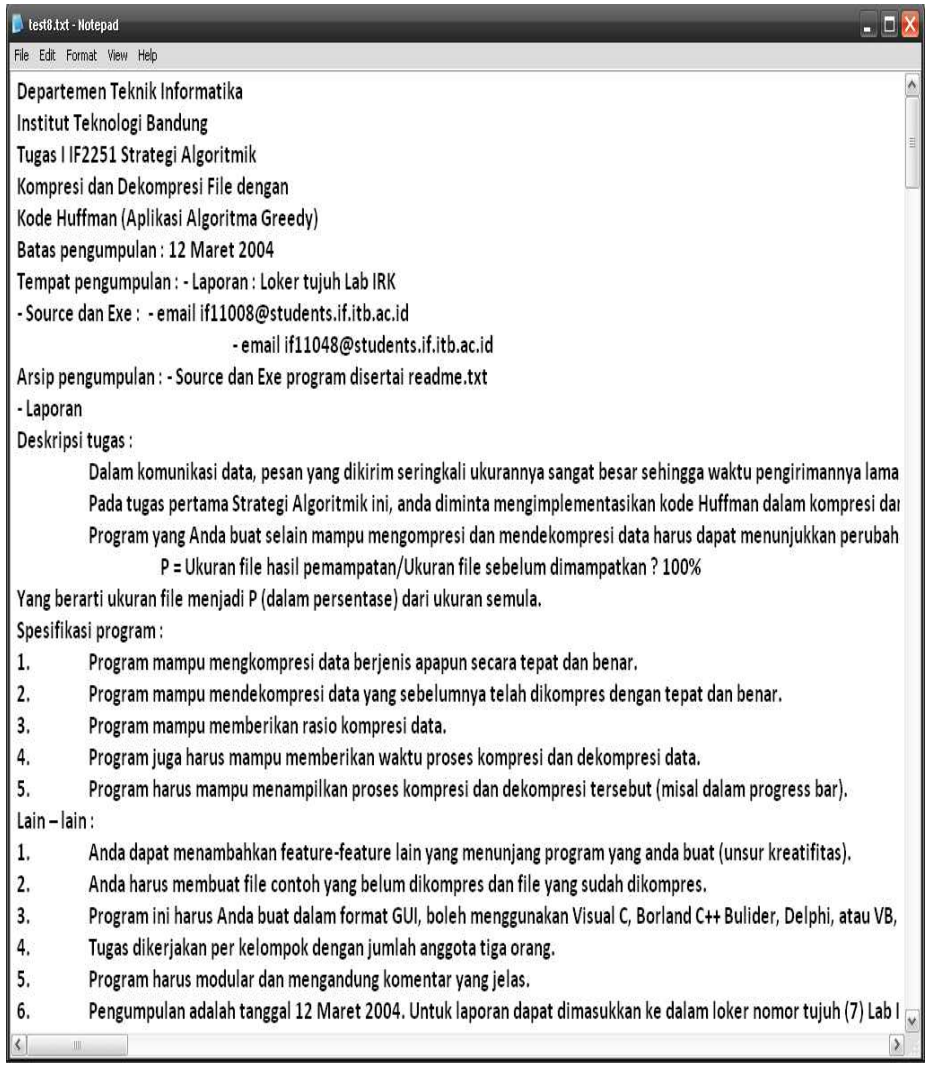

test6.txt - Notepad
File Edit Format View Help
Berikut protokol-protokol dalam layer-layer di dalam stack protokol bluetooth yang tertera pada Tabel 3.
Tabel 3. Protokol-protokol dan layer-layer di stack protokol bluetooth (sumber :
Bluetooth SIG)
Koneksi bluetooth dilakukan melalui beberapa tehnik berikut :
standby : device yang tidak terhubung kedalam sebuah piconet adalah termasuk dalam mode standby. Pada mode ini, mereka
page/ inquiry : jika sebuah device ingin melakukan sebuah proses koneksi dengan device lainnya, ia akan mengirmkan sebuah
active : transmisi antar data terjadi.
hold :
sniff : mode sniff, hanya etrsedia untuk unit slave,
Pengukuran Bluetooth
Pada dasarnya ada tiga aspek penting didalam melakukan pengukuran Bluetooth yaitu pengukuran RF (Radio Frequency), pro
Dari informasi Test & Measurement World, untuk pengukuran protokol, dapat menggunakan protocol sniffer yang dapat men
Contoh :
LAN access profile menentukan bagaimana perangkat bluetooth mampu mengakses layanan-layanan pada sebuah LAN meng
Fax profile menentukan persyaratan-persyaratan perangkat bluetooth yang harus dipenuhi untuk dapat mendukung layanan
Fungsi Sekurity
Bluetooth dirancang untuk memiliki fitur-fitur keamanan sehingga dapat digunakan secara aman baik dalam lingkungan bisnis
v Enkripsi data.
v Autentikasi user
v Fast frekuensi-hopping (1600 hops/sec)
v Output power control
Fitur-fitur tersebut menyediakan fungsi-fungsi keamanan dari tingkat keamanan layer fisik/ radio yaitu gangguan dari penyac

## 7. Test 7


test7.txt - Notepad
File Edit Format View Help
(erabaru.or.id) Pada jaman dahulu, seorang nenek tua menganut kepercayaan pada Dewi Kwan-im, ia selalu membaca kitab s
Si wanita tua yang menghadang jalan itu berkata, "Tiap hari merindukan saya, memuja saya, dan bertemu dengan saya lalu tic
Sebagian besar orang di dunia sekarang ini seperti anak kecil yang kebingungan, tersesat, mengembara ke mana-mana, mena
Setiap hari penganut Buddha memohon penyelamatan, berharap Buddha menetap di dunia, tidak tahu sudah berapa kehidup
(Sumber: Dajiyuan)
9. File 9
Departemen Teknik Informatika
Institut Teknologi Bandung
Tugas I IF2251 Strategi Algoritmik
Kompresi dan Dekompresi File dengan
Kode Huffman (Aplikasi Algoritma Greedy)
Batas pengumpulan : 12 Maret 2004
Tempat pengumpulan : - Laporan : Loker tujuh Lab IRK
- Source dan Exe : - email if11008@students.if.itb.ac.id
- email if11048@students.if.itb.ac.id
Arsip pengumpulan : - Source dan Exe program disertai readme.txt
- Laporan
Deskripsi tugas :
Dalam komunikasi data, pesan yang dikirim seringkali ukurannya sangat besar sehingga waktu pengirimannya lama
Pada tugas pertama Strategi Algoritmik ini, anda diminta mengimplementasikan kode Huffman dalam kompresi da
Program yang Anda buat selain mampu mengompresi dan mendekompresi data harus dapat menunjukkan perubah
P = Ukuran file hasil pemampatan/Ukuran file sebelum dimampatkan ? 100%
Yang berarti ukuran file menjadi P (dalam persentase) dari ukuran semula.
Spesifikasi program :
1. Program mampu mengkompresi data berjenis apapun secara tepat dan benar.
2. Program mampu mendekompresi data yang sebelumnya telah dikompres dengan tepat dan benar.
3. Program mampu memberikan rasio kompresi data.
4. Program juga harus mampu memberikan waktu proses kompresi dan dekompresi data.
5. Program harus mampu menampilkan proses kompresi dan dekompresi tersebut (misal dalam progress bar).
Lain – lain :


test7.txt - Notepad
File Edit Format View Help
- Flood fill : fills a connected region of a multi-dimensional array with a specified symbol
- Painter's algorithm : detects visible parts of a 3-dimensional scenery
- Ray tracing : realistic image rendering (computer graphics)
Algoritma Kriptografi
Lihat juga Topik dalam kriptografi
- symmetric key algorithm :
- Advanced Encryption Standard (AES), winner of NIST competition
- Blowfish (cipher)
- Data Encryption Standard (DES), sometimes DE Algorithm, winner of NBS selection competition, replaced by AES for most
- International Data Encryption Algorithm
- RC4 (cipher)
- Asymmetric key algorithm or digital signature :
- Digital Signature Algorithm
- ElGamal encryption
- RSA
- Diffie-Hellman key exchange
- NTRUEncrypt
- Cryptographic Message digest functions:
- MD5 – Note that there is now a method of generating collisions for MD5
- RIPEMD-160
- SHA-1
- keyed-hash message authentication code : keyed-hash message authentication
- Cryptographically secure pseudo-random number generator s
- Blum Blum Shub - based on the hardness of integer factorization
- Yarrow algorithm
- Fortuna (PRNG) , allegedly an improvement on Yarrow
- Other
- Diffie-Hellman : key exchange
Algoritma Distributed systems

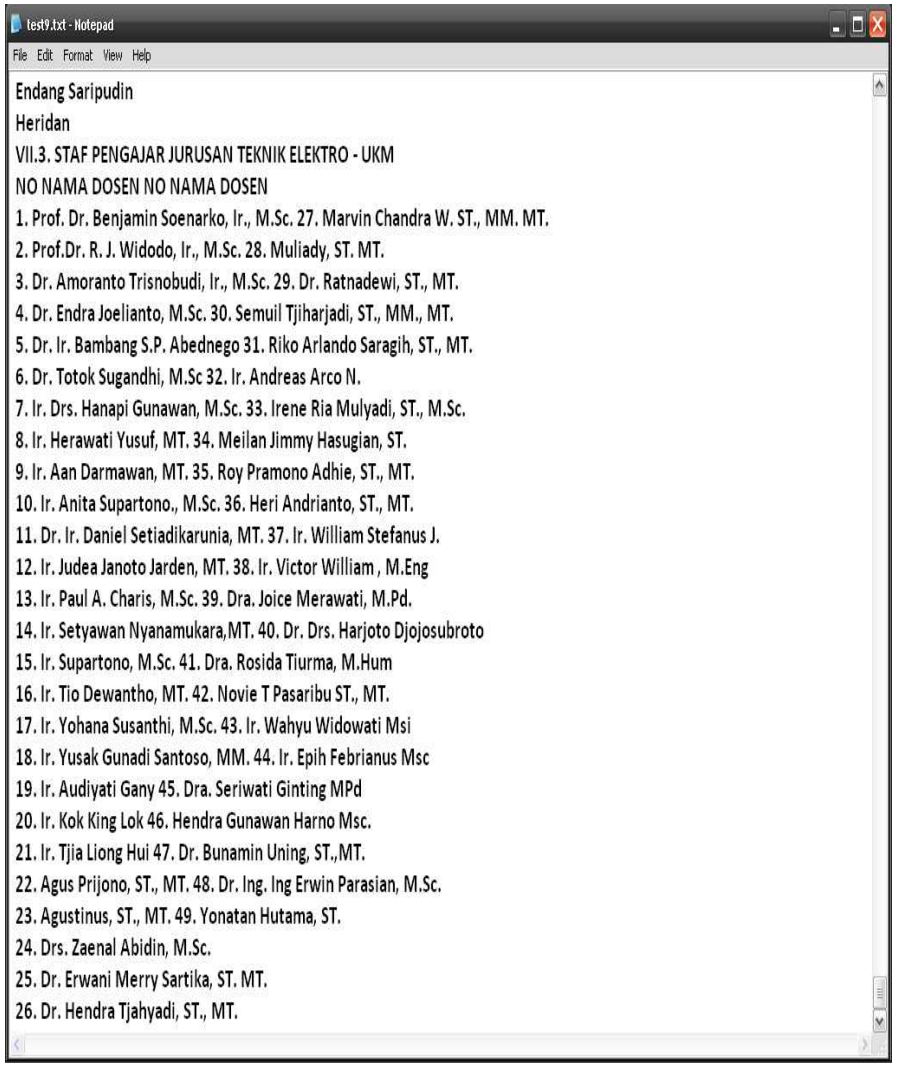
## 8. Test 8


test8.txt - Notepad
File Edit Format View Help
Departemen Teknik Informatika
Institut Teknologi Bandung
Tugas I IF2251 Strategi Algoritmik
Kompresi dan Dekompresi File dengan
Kode Huffman (Aplikasi Algoritma Greedy)
Batas pengumpulan : 12 Maret 2004
Tempat pengumpulan : - Laporan : Loker tujuh Lab IRK
- Source dan Exe : - email if11008@students.if.itb.ac.id
- email if11048@students.if.itb.ac.id
Arsip pengumpulan : - Source dan Exe program disertai readme.txt
- Laporan
Deskripsi tugas :
Dalam komunikasi data, pesan yang dikirim seringkali ukurannya sangat besar sehingga waktu pengirimannya lama
Pada tugas pertama Strategi Algoritmik ini, anda diminta mengimplementasikan kode Huffman dalam kompresi dar
Program yang Anda buat selain mampu mengompresi dan mendekompresi data harus dapat menunjukkan perubah
P = Ukuran file hasil pemampatan/Ukuran file sebelum dimampatkan ? 100%
Yang berarti ukuran file menjadi P (dalam persentase) dari ukuran semula.
Spesifikasi program :
1. Program mampu mengkompresi data berjenis apapun secara tepat dan benar.
2. Program mampu mendekompresi data yang sebelumnya telah dikompres dengan tepat dan benar.
3. Program mampu memberikan rasio kompresi data.
4. Program juga harus mampu memberikan waktu proses kompresi dan dekompresi data.
5. Program harus mampu menampilkan proses kompresi dan dekompresi tersebut (misal dalam progress bar).
Lain – lain :
1. Anda dapat menambahkan feature-feature lain yang menunjang program yang anda buat (unsur kreatifitas).
2. Anda harus membuat file contoh yang belum dikompres dan file yang sudah dikompres.
3. Program ini harus Anda buat dalam format GUI, boleh menggunakan Visual C, Borland C++ Bulider, Delphi, atau VB,
4. Tugas dikerjakan per kelompok dengan jumlah anggota tiga orang.
5. Program harus modular dan mengandung komentar yang jelas.
6. Pengumpulan adalah tanggal 12 Maret 2004. Untuk laporan dapat dimasukkan ke dalam loker nomor tujuh (7) Lab I

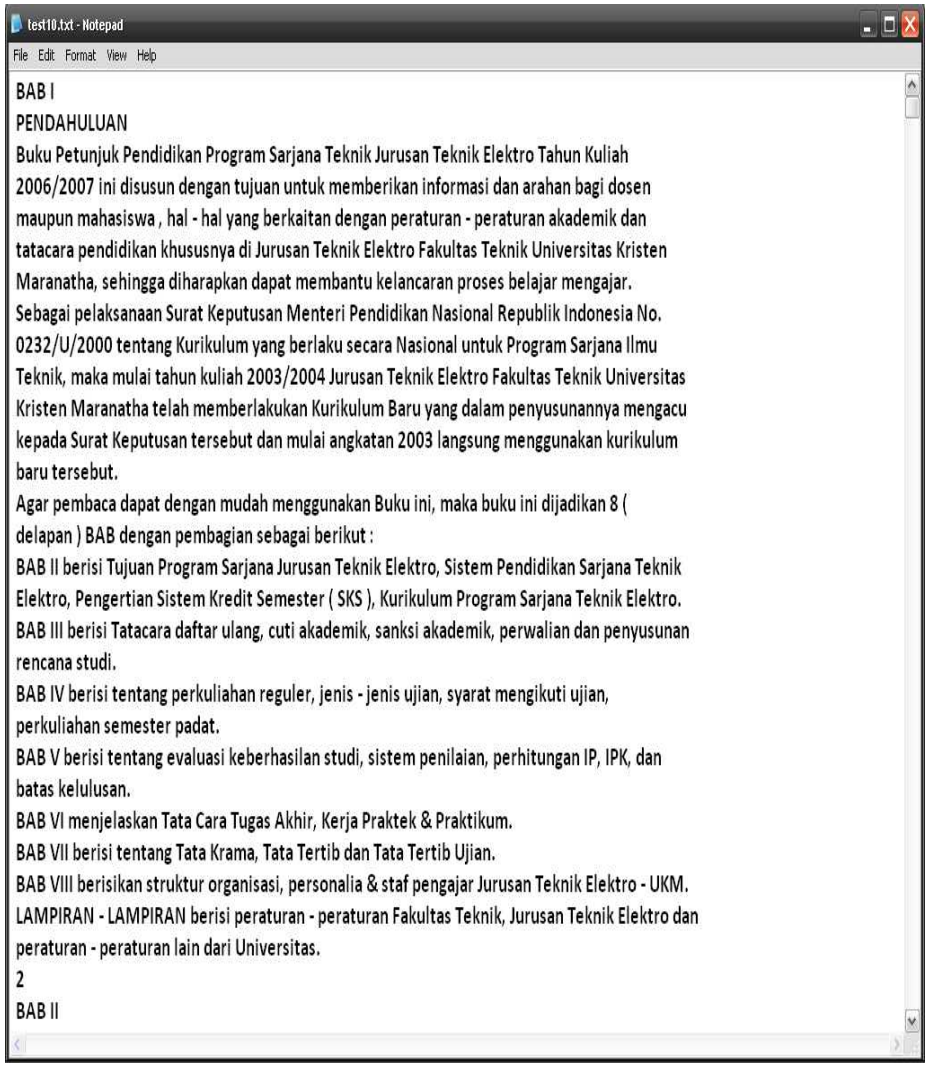

test8.txt - Notepad
File Edit Format View Help
- Kenneth Iverson Penemu APL, untuk kontribusinya di perhitungan interaktif.
- William Kahan untuk standard IEEE floating-point.
- Donald Knuth untuk Seni dari Pemrograman Komputer
- Ada Lovelace programer terkenal pertama di dunia
- John von Neumann yang telah mengembangkan arsitektur von Neumann .
- Claude E. Shannon untuk teori informasi
- Alan Turing untuk teori komputabilitas.
- James Wilkinson Teknik "analisa kesalahan dari belakang" dan kemajuan di bidang perhitungan matriks. Wilkinson adalah ju
- Konrad Zuse Pembuat binari komputer yang pertama pada 1930-an, di mana dia menrencanakan bahasa pemrograman jauh
Lihat juga
- Bug
- Bahasa Pemrograman
- Perhitungan
- Sejarah dari Perhitungan
- Turing Award ( ACM )
- Medali IEEE John von Neumann
- Hello world
- Istilah Komputer
- Berkas Istilah
- Topik utama Ilmu Komputer
- Analogi Perhitungan
- Internet
- Multimedia
- Akusisi data
- Tolok
- Jaringan Sensor
- Komputasi dan Algorithma Online ,
- Format Bilangan Komputer
Pranala Luar

## 9. Test 9


test9.txt - Notepad
File Edit Format View Help
0
BAB III
DAFTAR ULANG & PERWALIAN
III.1. DAFTAR ULANG
a. Setiap mahasiswa wajib melakukan pendaftaran ulang ( herregistrasi ) pada setiap awal semester. Mahasiswa yang cuti studi maupun yang terkena skorsing tetap diwajibkan melakukan daftar ulang dengan hanya membayar biaya daftar ulang.
b. Pendaftaran Ulang harus disertai dengan pembayaran :
 Biaya kuliah sejumlah SKS yang diambil dengan minimal adalah 12 SKS kecuali bagi mahasiswa yang memang pada semester tersebut tersisa kurang dari 12 SKS ( dengan surat keterangan dari dosen wali mahasiswa yang bersangkutan )
 Biaya Pengembangan
 Biaya Daftar Ulang
 Biaya Asuransi
Besarnya masing - masing biaya sesuai dengan ketentuan yang ditetapkan oleh Universitas.
c. Mahasiswa dapat pula melunasi kewajiban keuangan dalam 2 tahap ( angsuran ) dengan ketentuan sebagai beriut :
 Angsuran I : Biaya kuliah minimal 5 SKS, Biaya pengembangan, Biaya Daftar Ulang, Biaya Asuransi.
 Angsuran II : Sisa biaya kuliah yang belum dilunasi.
d. Mahasiswa yang tidak melakukan Daftar Ulang atau sedang cuti akademik atau juga terkena skorsing tidak berhak mengikuti seluruh kegiatan akademik ( kuliah, praktikum, ujian, bimbingan dan lain sebagainya ) serta menggunakan fasilitas yang ada ( peminjaman buku perpustakaan, laboratorium dan sebagainya ) pada semester tersebut.
III.2. REGISTRASI MAHASISWA BARU
Pendaftaran mahasiswa baru di Jurusan Teknik Elektro - UKM dilaksanakan pada awal tahun akademik dengan syarat :
1. Dinyatakan lulus/ diterima menjadi mahasiswa Jurusan Teknik Elektro - UKM.
2. Mengisi dan menyerahkan kembali Formulir Kartu Mahasiswa Jurusan Teknik Elektro dengan menyerahkan foto copy bukti pembayaran uang kuliah.


test9.txt - Notepad
File Edit Format View Help
Endang Saripudin
Heridan
VII.3. STAF PENGAJAR JURUSAN TEKNIK ELEKTRO - UKM
NO NAMA DOSEN NO NAMA DOSEN
1. Prof. Dr. Benjamin Soenarko, Ir., M.Sc. 27. Marvin Chandra W. ST., MM. MT.
2. Prof.Dr. R. J. Widodo, Ir., M.Sc. 28. Muliady, ST. MT.
3. Dr. Amoranto Trisnobudi, Ir., M.Sc. 29. Dr. Ratnadewi, ST., MT.
4. Dr. Endra Joelianto, M.Sc. 30. Semuil Tjiharjadi, ST., MM., MT.
5. Dr. Ir. Bambang S.P. Abednego 31. Riko Arlando Saragih, ST., MT.
6. Dr. Totok Sugandhi, M.Sc 32. Ir. Andreas Arco N.
7. Ir. Drs. Hanapi Gunawan, M.Sc. 33. Irene Ria Mulyadi, ST., M.Sc.
8. Ir. Herawati Yusuf, MT. 34. Meilan Jimmy Hasugian, ST.
9. Ir. Aan Darmawan, MT. 35. Roy Pramono Adhie, ST., MT.
10. Ir. Anita Supartono., M.Sc. 36. Heri Andrianto, ST., MT.
11. Dr. Ir. Daniel Setiadikarunia, MT. 37. Ir. William Stefanus J.
12. Ir. Judea Janoto Jarden, MT. 38. Ir. Victor William , M.Eng
13. Ir. Paul A. Charis, M.Sc. 39. Dra. Joice Merawati, M.Pd.
14. Ir. Setyawan Nyanamukara,MT. 40. Dr. Drs. Harjoto Djojosubroto
15. Ir. Supartono, M.Sc. 41. Dra. Rosida Tiurma, M.Hum
16. Ir. Tio Dewantho, MT. 42. Novie T Pasaribu ST., MT.
17. Ir. Yohana Susanthi, M.Sc. 43. Ir. Wahyu Widowati Msi
18. Ir. Yusak Gunadi Santoso, MM. 44. Ir. Epih Febrianus Msc
19. Ir. Audiyati Gany 45. Dra. Seriwati Ginting MPd
20. Ir. Kok King Lok 46. Hendra Gunawan Harno Msc.
21. Ir. Tjia Liong Hui 47. Dr. Bunamin Uning, ST.,MT.
22. Agus Prijono, ST., MT. 48. Dr. Ing. Ing Erwin Parasian, M.Sc.
23. Agustinus, ST., MT. 49. Yonatan Hutama, ST.
24. Drs. Zaenal Abidin, M.Sc.
25. Dr. Erwani Merry Sartika, ST. MT.
26. Dr. Hendra Tjahyadi, ST., MT.

## 10. Test 10


test10.txt - Notepad
File Edit Format View Help
BAB I
PENDAHULUAN
Buku Petunjuk Pendidikan Program Sarjana Teknik Jurusan Teknik Elektro Tahun Kuliah 2006/2007 ini disusun dengan tujuan untuk memberikan informasi dan arahan bagi dosen maupun mahasiswa , hal - hal yang berkaitan dengan peraturan - peraturan akademik dan tatacara pendidikan khususnya di Jurusan Teknik Elektro Fakultas Teknik Universitas Kristen Maranatha, sehingga diharapkan dapat membantu kelancaran proses belajar mengajar.
Sebagai pelaksanaan Surat Keputusan Menteri Pendidikan Nasional Republik Indonesia No. 0232/U/2000 tentang Kurikulum yang berlaku secara Nasional untuk Program Sarjana Ilmu Teknik, maka mulai tahun kuliah 2003/2004 Jurusan Teknik Elektro Fakultas Teknik Universitas Kristen Maranatha telah memberlakukan Kurikulum Baru yang dalam penyusunannya mengacu kepada Surat Keputusan tersebut dan mulai angkatan 2003 langsung menggunakan kurikulum baru tersebut.
Agar pembaca dapat dengan mudah menggunakan Buku ini, maka buku ini dijadikan 8 ( delapan ) BAB dengan pembagian sebagai berikut :
BAB II berisi Tujuan Program Sarjana Jurusan Teknik Elektro, Sistem Pendidikan Sarjana Teknik Elektro, Pengertian Sistem Kredit Semester ( SKS ), Kurikulum Program Sarjana Teknik Elektro.
BAB III berisi Tatacara daftar ulang, cuti akademik, sanksi akademik, perwalian dan penyusunan rencana studi.
BAB IV berisi tentang perkuliahan reguler, jenis - jenis ujian, syarat mengikuti ujian, perkuliahan semester padat.
BAB V berisi tentang evaluasi keberhasilan studi, sistem penilaian, perhitungan IP, IPK, dan batas kelulusan.
BAB VI menjelaskan Tata Cara Tugas Akhir, Kerja Praktek & Praktikum.
BAB VII berisi tentang Tata Krama, Tata Tertib dan Tata Tertib Ujian.
BAB VIII berisikan struktur organisasi, personalia & staf pengajar Jurusan Teknik Elektro - UKM.
LAMPIRAN - LAMPIRAN berisi peraturan - peraturan Fakultas Teknik, Jurusan Teknik Elektro dan peraturan - peraturan lain dari Universitas.
2
BAB II


test10.txt - Notepad
File Edit Format View Help
dan catu daya yang diatur.
Prasyarat : TE-235
Referensi :
1. Millman-Halkias, "Integrated Electronics", McGraw Hill, 1985.
2. Price, "Analog Electronics", Prentice Hall, 1997
Semester : 5 Kode MK : TE - 311
Nama MK : PENGOLAHAN SINYAL DIJITAL 3 SKS
Kuliah ini membahas tentang elemen dasar dan struktur sistem pengolahan sinyal dijital, konsep frekuensi sinyal waktu kontinyu dan diskrit, analisis sistem linier time-invariant, perancangan filter IIR dan FIR dengan metoda-metoda impuls-invariant, transformasi bilinier, least squres, transformasi frekuensi dalam domain dijital, window, frekuensi cuplik, aplikasi DFT, korelasi sendiri dan korelasi silang.
Prasyarat : TE-226
Referensi :
1. Ludeman, "Fundamentals of Digital Signal Processing", John Wiley & Sons, 1987.
2. Oppenheim & Schafer, "Discrete-time Signal Processing", Prentice Hall, 1991.
3. Proakis & Manolakis. "Digital Signal Processing", Prentice Hall, 1996.
Semester : 5 Kode MK : TE - 341
Nama MK : SALURAN TRANSMISI 3 SKS
Kuliah ini membahas saluran-saluran (seperti kawat, kabel coax, wave guide dan fiber optic) yang digunakan untuk transmisi sinyal informasi dan melakukan analisis terhadap saluran-saluran tersebut secara analitis dan grafis yaitu dengan mengunakan Smith Chart. Dibahas juga penyesuaian impedansi dengan Stub tunggal dan ganda.
Prasyarat : TE-178
Referensi :
1. Johnson, "Transmission Lines and Networks"
2. Skilling, "Transmission Lines"
Semester : 6 Kode MK : TE - 362
Nama MK : JARINGAN TELEKOMUNIKASI 3 SKS
Kuliah ini membahas tentang jaringan yang digunakan untuk komunikasi meliputi jaringan