Kelompok 7 :

Ian Felix Jonathan	: 05311940000008
Shavica Ihya Qurata Ayun L	: 05311940000013
Daniel Evan Yudhiputra	: 05311940000016
Bryan Yehuda Mannuel	: 05311940000021
Revina Rahmanisa Harjanto	: 05311940000046

Laporan ETS Teknologi Multimedia Dictionary Based

Soal :

Membuat implementasi program dengan 2 algoritma teks-image menggunakan metode Shannon Fano-Dictionary based dalam komunikasi client-server dan hitung rasio kompresi setiap algoritma.

Jawaban :

Kami menggunakan

Dictionary based (Image)

Pada directory based ini kami menggunakan algoritma LZW (Lempel-Ziv-Welch). Berikut adalah source code yang kami gunakan

Client :

```
#!/usr/bin/env python

import random
import socket, select
from time import gmtime, strftime
from random import randint
import time
import os

#image = raw_input("enter the name of the image file: ")

HOST = '127.0.0.1'
PORT = 6662

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_address = (HOST, PORT)
```

```
sock.connect(server_address)

try:
      os.system('python3 LZW.py')
      time.sleep(20)
      myfile =
open('/home/daniel/Documents/cmpr/CompressedFiles/checkCompressed.
lzw', 'rb')
      bytes = myfile.read()
      size = len(bytes)

      # send image size to server
      sock.sendall("SIZE %s" % size)
      answer = sock.recv(4096)

      print 'answer = %s' % answer

      # send image to server
      if answer == 'GOT SIZE':
      sock.sendall(bytes)

      # check what server send
      answer = sock.recv(4096)
      print 'answer = %s' % answer

      if answer == 'GOT IMAGE' :
            #time.sleep(20)
            sock.sendall("BYE BYE ")
            print 'File successfully send to server'
            file_size1 =
float(os.path.getsize('/home/daniel/Documents/cmpr/CompressedFiles
/checkCompressed.lzw'))
            file_size2 = float(os.path.getsize('check.tif'))
            ukuran = float(file_size2/file_size1)
            print "rasio kompresi : "+str(float(ukuran))

      myfile.close()

finally:
      sock.close()
```

Berikut adalah penjelasan dari code diatas :

1. Library

```
import random
import socket, select
from time import gmtime, strftime
from random import randint
import time
import os
```

Random digunakan untuk melakukan generate random number, socket untuk membuat socket, time untuk mendapatkan time, time untuk melakukan fungsi sleep, dan os untuk melakukan fungsi system.

2. Membuat socket

```
HOST = '127.0.0.1'
PORT = 6662

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_address = (HOST, PORT)
sock.connect(server_address)
```

Fungsi diatas digunakan untuk membuat socket pada local.

3. Memanggil program LZW

```
os.system('python3 LZW.py')
time.sleep(20)
```

Fungsi diatas digunakan untuk memanggil program LZW, dan melakukan sleep 20 detik, untuk memastikan bahwa program telah selesai di run.

4. Mengirim size ke server

```
myfile = 
open('/home/daniel/Documents/cmpr/CompressedFiles/checkCompre
ssed.lzw', 'rb')
bytes = myfile.read()
size = len(bytes)

# send image size to server
sock.sendall("SIZE %s" % size)
answer = sock.recv(4096)

print 'answer = %s' % answer
```

Fungsi diatas digunakan untuk membuka file dalam bentuk biner dan mengirim size dari file ke server.

5. Mengirim file ke server

```
# send image to server
      if answer == 'GOT SIZE':
      sock.sendall(bytes)

      # check what server send
      answer = sock.recv(4096)
      print 'answer = %s' % answer
```

Fungsi diatas digunakan untuk mengirim file ke server.

6. Mengukur rasio kompresi

```
           file_size1 =
float(os.path.getsize('/home/daniel/Documents/cmpr/Compressed
Files/checkCompressed.lzw'))
           file_size2 = float(os.path.getsize('check.tif'))
           ukuran = float(file_size2/file_size1)
           print "rasio kompresi : "+str(float(ukuran))
```

Fungsi diatas digunakan untuk menghitung rasio kompresi dengan mendapat ukuran file sebelum kompresi dibagi ukuran file setelah kompresi.

7. Menutup socket

```
finally:
      sock.close()
```

Fungsi diatas digunakan untuk menutup socket.

Server :

```
#!/usr/bin/env python

import random
import socket, select
from time import gmtime, strftime
from socket import error as SocketError
from random import randint
basename = "checkCompressed.lzw"
import os
import errno
import time

imgcounter = 1



HOST = '127.0.0.1'
```

```
PORT = 6662

connected_clients_sockets = []

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

server_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR,
1)
server_socket.bind((HOST, PORT))
server_socket.listen(10)

connected_clients_sockets.append(server_socket)

while True:

    read_sockets, write_sockets, error_sockets =
select.select(connected_clients_sockets, [], [])

    for sock in read_sockets:
    if sock == server_socket:

        sockfd, client_address = server_socket.accept()
        connected_clients_sockets.append(sockfd)

    else:

        try:
            data = sock.recv(4096)
            txt = str(data)

            if data:

            if data.startswith('SIZE'):
                tmp = txt.split()
                size = int(tmp[1])

                print 'got size'

                sock.sendall("GOT SIZE")

            elif data.startswith('BYE'):
                sock.shutdown()
            else :

                myfile = open(basename, 'wb')
                myfile.write(data)

                data = sock.recv(40960000)
                if not data:
                     myfile.close()
                     break
                myfile.write(data)
                time.sleep(20)
                myfile.close()

                sock.sendall("GOT IMAGE")
                sock.shutdown(0)
        except SocketError as e:
            if e.errno != errno.ECONNRESET:
                raise
            pass
    imgcounter += 1
server_socket.close()
```

Berikut adalah penjelasan dari code diatas :

1. Library yang digunakan

```
import random
import socket, select
from time import gmtime, strftime
from socket import error as SocketError
from random import randint
import os
import errno
import time
```

   Random digunakan untuk melakukan generate random number, socket untuk membuat socket, time untuk mendapatkan waktu saat ini SocketError dan errno untuk menghandle error socket, os untuk menjalankan fungsi system() dan time untuk fungsi sleep()

2. Membuat socket

```
HOST = '127.0.0.1'
PORT = 6662
```

```
connected_clients_sockets = []

server_socket = socket.socket(socket.AF_INET,
socket.SOCK_STREAM)

server_socket.setsockopt(socket.SOL_SOCKET,
socket.SO_REUSEADDR, 1)
server_socket.bind((HOST, PORT))
server_socket.listen(10)

connected_clients_sockets.append(server_socket)

while True:
     read_sockets, write_sockets, error_sockets =
select.select(connected_clients_sockets, [], [])

     for sock in read_sockets:
     if sock == server_socket:
          sockfd, client_address = server_socket.accept()
          connected_clients_sockets.append(sockfd)
```

Fungsi diatas digunakan untuk membuat socket.

3. Mendapat size

```
try:
                data = sock.recv(4096)
                txt = str(data)

                if data:

                if data.startswith('SIZE'):
                     tmp = txt.split()
                     size = int(tmp[1])

                     print 'got size'

                     sock.sendall("GOT SIZE")
```

Fungsi diatas digunakan untuk mendapatkan ukuran file yang dikirimkan oleh client.

4. Mendapat file

```
myfile = open(basename, 'wb')
myfile.write(data)
```

```
data = sock.recv(40960000)
if not data:
   myfile.close()
   break
myfile.write(data)
time.sleep(20)
myfile.close()

sock.sendall("GOT IMAGE")
sock.shutdown(0)
```

Fungsi di atas digunakan untuk mendapat file dari client dan menyimpannya dalam sebuah file lain

5. Socket error handling

```
except SocketError as e:
if e.errno != errno.ECONNRESET:
    raise
pass
```

Fungsi diatas digunakan untuk menangani error pada socket agar program tidak terminate.

LZW :

```
import LZW
from PIL import Image
import os
import numpy as np

class LZW:
     def __init__(self, path):
     self.path = path
     self.compressionDictionary, self.compressionIndex =
self.createCompressionDict()
     self.decompressionDictionary, self.decompressionIndex =
self.createDecompressionDict()

     def compress(self):
     self.initCompress()
     compressedcColors = []
     #print("Compressing Image ...")
     compressedcColors.append(self.compressColor(self.red))
     #print("Compressing Image ...")
     compressedcColors.append(self.compressColor(self.green))
     #print("Compressing Image ...")
     compressedcColors.append(self.compressColor(self.blue))
     #print("Image Compressed --------- Writing to File")
     filesplit = str(os.path.basename(self.path)).split('.')
     filename = filesplit[0] + 'Compressed.lzw'
     savingDirectory = os.path.join(os.getcwd(),'CompressedFiles')
     if not os.path.isdir(savingDirectory):
          os.makedirs(savingDirectory)
     with open(os.path.join(savingDirectory,filename),'w') as
file:
          for color in compressedcColors:
               for row in color:
               file.write(row)
               file.write("\n")

     def compressColor(self, colorList):
     compressedColor = []
     i = 0
     for currentRow in colorList:
          currentString = currentRow[0]
          compressedRow = ""
          i+=1
          for charIndex in range(1, len(currentRow)):
               currentChar = currentRow[charIndex]
               if currentString+currentChar in
self.compressionDictionary:
               currentString = currentString+currentChar
               else:
               compressedRow = compressedRow +
str(self.compressionDictionary[currentString]) + ","

     self.compressionDictionary[currentString+currentChar] =
self.compressionIndex
               self.compressionIndex += 1
               currentString = currentChar
               currentChar = ""
          compressedRow = compressedRow +
str(self.compressionDictionary[currentString])
          compressedColor.append(compressedRow)
```

```
    return compressedColor

    def initCompress(self):
    self.image = Image.open(self.path)
    self.height, self.width = self.image.size
    self.red, self.green, self.blue = self.processImage()

    def processImage(self):
    image = self.image.convert('RGB')
    red, green, blue = [], [], []
    pixel_values = list(image.getdata())
    iterator = 0
    for height_index in range(self.height):
        R, G, B = "","",""
        for width_index in range(self.width):
            RGB = pixel_values[iterator]
            R = R + str(RGB[0]) + ","
            G = G + str(RGB[1]) + ","
            B = B + str(RGB[2]) + ","
            iterator+=1
        red.append(R[:-1])
        green.append(G[:-1])
        blue.append(B[:-1])
    return red,green,blue

    def createDecompressionDict(self):
    dictionary = {}
    for i in range(10):
        dictionary[i] = str(i)
    dictionary[10] = ','
    return dictionary,11

compressor = LZW('check.tif')
compressor.compress()
```

Berikut adalah penjelasan code diatas :

1. Library

```
import LZW
from PIL import Image
import os
import numpy as np
```

LZW untuk menjalankan algoritma LZW, PIL untuk menghandle image dan memecahnya menjadi rgb, os untuk system, dan numpy untuk image

2. Fungsi compress

```
def compress(self):
    self.initCompress()
    compressedcColors = []
    #print("Compressing Image ...")
    compressedcColors.append(self.compressColor(self.red))
    #print("Compressing Image ...")
    compressedcColors.append(self.compressColor(self.green))
    #print("Compressing Image ...")
    compressedcColors.append(self.compressColor(self.blue))
    #print("Image Compressed --------- Writing to File")
    filesplit = str(os.path.basename(self.path)).split('.')
    filename = filesplit[0] + 'Compressed.lzw'
    savingDirectory =
os.path.join(os.getcwd(),'CompressedFiles')
    if not os.path.isdir(savingDirectory):
        os.makedirs(savingDirectory)
    with open(os.path.join(savingDirectory,filename),'w') as
file:
        for color in compressedcColors:
            for row in color:
            file.write(row)
            file.write("\n")
```

Fungsi diatas secara umum akan memanggil fungsi-fungsi lainnya untuk berjalan.

3. Fungsi compress color

```
def compressColor(self, colorList):
    compressedColor = []
    i = 0
    for currentRow in colorList:
        currentString = currentRow[0]
        compressedRow = ""
        i+=1
        for charIndex in range(1, len(currentRow)):
            currentChar = currentRow[charIndex]
            if currentString+currentChar in
self.compressionDictionary:
            currentString = currentString+currentChar
            else:
            compressedRow = compressedRow +
str(self.compressionDictionary[currentString]) + ","

    self.compressionDictionary[currentString+currentChar] =
self.compressionIndex
            self.compressionIndex += 1
            currentString = currentChar
            currentChar = ""
        compressedRow = compressedRow +
str(self.compressionDictionary[currentString])
        compressedColor.append(compressedRow)
    return compressedColor
```

Fungsi diatas digunakan untuk melakukan compress pada file gambar dengan melakukan compress warna yang akan dipanggil di fungsi lainnya.

4. Fungsi initial compress

```
def initCompress(self):
    self.image = Image.open(self.path)
    self.height, self.width = self.image.size
    self.red, self.green, self.blue = self.processImage()
```

Fungsi ini akan membuka file image, mendapat ukuran image, dan mendapatkan nilai RGB dari image

5. Fungsi image process

```
def processImage(self):
    image = self.image.convert('RGB')
    red, green, blue = [], [], []
    pixel_values = list(image.getdata())
    iterator = 0
    for height_index in range(self.height):
        R, G, B = "","",""
        for width_index in range(self.width):
            RGB = pixel_values[iterator]
            R = R + str(RGB[0]) + ","
            G = G + str(RGB[1]) + ","
            B = B + str(RGB[2]) + ","
            iterator+=1
        red.append(R[:-1])
        green.append(G[:-1])
        blue.append(B[:-1])
    return red,green,blue
```

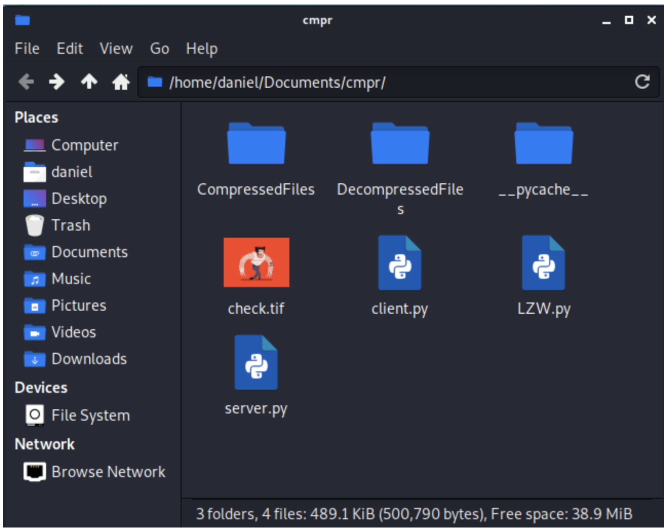
Pada fungsi ini nilai dari tiap RGB akan dipisahkan.

6. Fungsi create compression directory

```
def createCompressionDict(self):
    dictionary = {}
    for i in range(10):
        dictionary[str(i)] = i
    dictionary[','] = 10
    return dictionary,11
```

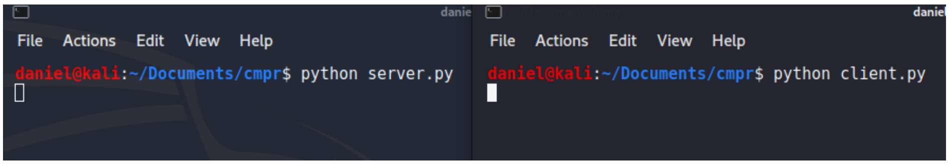
Fungsi diatas digunakan untuk membuat dictionary pada proses compress.

Hasil Run :



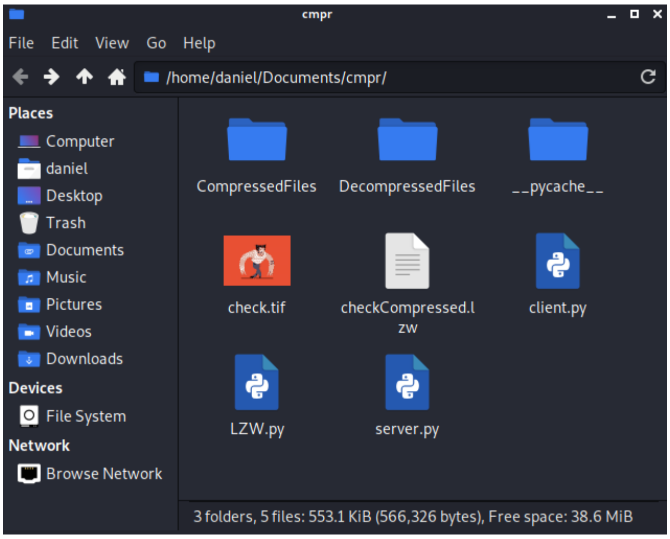
/home/daniel/Documents/cmpr/

3 folders, 4 files: 489.1 KiB (500,790 bytes), Free space: 38.9 MiB

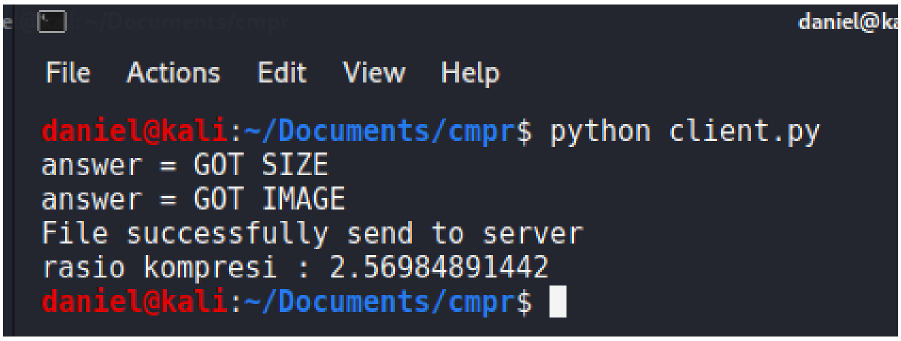


```
daniel@kali:~/Documents/cmpr$ python server.py
```

```
daniel@kali:~/Documents/cmpr$ python client.py
```



3 folders, 5 files: 553.1 KiB (566,326 bytes), Free space: 38.6 MiB



```
daniel@kali:~/Documents/cmpr$ python client.py
answer = GOT SIZE
answer = GOT IMAGE
File successfully send to server
rasio kompresi : 2.56984891442
daniel@kali:~/Documents/cmpr$
```