**LAPORAN TUGAS KECIL 4**

**IF2211 – STRATEGI ALGORITMA**

**EKSTRAKSI INFORMASI DARI ARTIKEL BERITA**

**DENGAN ALGORITMA PENCOCOKAN STRING**

****

Disusun oleh:

**Fikra Hadi Ramadhan / 13518036**

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**Jl. Ganeca No. 10, Bandung 40132**

**DAFTAR ISI**

[BAB 1 DESKRIPSI SINGKAT PENCOCOKAN STRING](#_Toc20459432) 3

[BAB 2 KODE PROGRAM](#_Toc20459441) 4

[BAB 3 SCREENSHOT INPUT-OUTPUT PROGRAM](#_Toc20459441) 16

[DAFTAR PUSTAKA](#_Toc20459444) 19

[PENILAIAN](#_Toc20459444) 20

# BAB 1 DESKRIPSI SINGKAT PENCOCOKAN STRING

1. Algoritma Knuth-Morris-Pratt

Algoritma Knuth Morris Pratt (KMP) dikembangkan oleh D. E. Knuth, bersama dengan J. H. Morris dan V. R. Pratt. Untuk pencarian string dengan menggunakan algoritma Brute Force , setiap kali ditemukan ketidakcocokan pattern dengan teks, maka pattern akan digeser satu karakter ke kanan. Sedangkan Algoritma Knuth Morris Pratt bekerja dengan memanfaatkan pergeseran yang semaksimal mungkin dalam pencocokan string dalam teks. padanannya dalam bahasa Inggris. Aplikasi penerjemah haruslah melakukan pencocokan kata untuk mencari kata tersebut dalam bahasa asing. Haruslah diperhitungkan apakah kata tersebut merupakan bagian dari kata majemuk lainnya.

Misalkan saja terdapat kata “kabar” yang ingin dicari

1. Algoritma Boyer-Moore

Boyer-Moore secara rata-rata merupakan algoritma pencarian string yang paling baik jika dibandingkan dengan algoritma pencarian string lainnya seperti Brute-Force ataupun Knuth-Morris-Pratt. Jika kita menggunakan fasilitas Find/Search pada berbagai aplikasi pengolah teks, web browser, dan aplikasi lainnya mungkin saja kita telah memanfaatkan algoritma Boyer-Moore dalam pencarian tersebut, karena algoritma ini paling banyak diimplementasikan dalam berbagai aplikasi untuk fasilitas pencarian teksnya.

Algoritma Boyer-Moore adalah salah satu algoritma untuk mencari suatu string di dalam teks, dibuat oleh R.M Boyer dan J.S Moore. Ide utama algoritma ini adalah mencari string dengan melakukan pembandingan karakter mulai dari karakter paling kanan dari string yang dicari.

1. Regular Expression

Sebagai seorang system administrators, developers, QA engineers, atau support engineers kita sering kali perlu mencari data dengan pola tertentu misalnya sekelompok IP addresses dalam kisaran tertentu atau terkadang kita juga perlu mencari kata yang salah tulis. Untuk kasus-kasus seperti ini, regular expression dapat membantu.

Regular expressions adalah template untuk data berdasarkan pola tertentu. Tutorial ini akan mendemonstrasikan kemampuan regular expression tanpa bergantung pada bahasa pemrograman tertentu dengan menggunakan egrep untuk menjelaskan konsepnya.

# BAB 2 KODE PROGRAM

* main.py

main program

# Nama : Fikra Hadi Ramadhan

# NIM : 13518036

# Hal : Tugas Kecil 4 - Strategi Algoritma

# Judul : Ekstraksi Informasi dari Artikel Berita dengan Algoritma Pencocokan String

import re

from os import listdir

from os.path import isfile, join

def kmpMatch(text, pattern):

n = len(text)

m = len(pattern)

fail = []

fail = computeFail(pattern)

i = 0

j = 0

while (i < n):

if (pattern[j] == text[i]):

if (j == m - 1):

return (i - m + 1) #cocok

i += 1

j += 1

elif (j > 0):

j = fail[j-1]

else:

i += 1

return -1 #tidak cocok

def computeFail(pattern):

fail = [0 for fail in range(len(pattern))]

fail[0] = 0

m = len(pattern)

j = 0

i = 1

while (i < m):

if (pattern[j] == pattern[i]): # cocok untuk j+1 chars

fail[i] = j + 1

i += 1

j += 1

elif (j > 0): #j mengikuti kecocokan prefix

j = fail[j-1]

else:

fail[i] = 0

i += 1

return fail

def mainKMP(text, keyword, filename, hasil2):

waktuArtikel = getWaktuArtikel(text)

low = text.lower()

pattern = keyword.lower()

posn = kmpMatch(low, pattern)

if (posn == -1):

print("Keyword not found")

else:

kalimat = kalimatRegex(text, keyword)

span = searchSpanPattern(kalimat, keyword)

waktu = waktuRegex(kalimat, waktuArtikel)

jumlah = jumlahRegex(kalimat, keyword, span)

for i in range(len(kalimat)):

data = {}

data['jumlah'] = jumlah[i]

data['waktu'] = waktu[i]

data['kalimat'] = kalimat[i]

data['filename'] = filename

hasil2.append(data)

def bmMatch(text, pattern):

last = buildLast(pattern)

n = len(text)

m = len(pattern)

i = m - 1

if (i > n-1):

return -1 #tidak match jika pattern lebih panjang dari text

j = m-1

while (i <= n-1):

if (pattern[j] == text[i]):

if (j == 0):

return i #cocok

else: #looking-glass technique

i -= 1

j -= 1

else: #character jump technique

lo = last[ord(text[i])] #last occ

i = i + m - min(j, 1+lo)

j = m - 1

if (i > n-1):

break

return -1 #tidak ada yang cocok

def buildLast(pattern):

last = [-1 for last in range(128)]

for i in range (len(pattern)):

last[ord(pattern[i])] = i

return last

def mainBM(text, keyword, filename, hasil2):

waktuArtikel = getWaktuArtikel(text)

low = text.lower()

pattern = keyword.lower()

posn = bmMatch(low, pattern)

if (posn == -1):

print("Keyword not found")

else:

kalimat = kalimatRegex(text, keyword)

span = searchSpanPattern(kalimat, keyword)

waktu = waktuRegex(kalimat, waktuArtikel)

jumlah = jumlahRegex(kalimat, keyword, span)

for i in range(len(kalimat)):

data = {}

data['jumlah'] = jumlah[i]

data['waktu'] = waktu[i]

data['kalimat'] = kalimat[i]

data['filename'] = filename

hasil2.append(data)

def jumlahRegex(kal, pattern, span):

x = len(kal)

jumlah = [0 for x in range(x)]

for i in range(x):

jumlahnya = []

jumlah\_regex = re.compile(r'(?:^|(?<=\s| ))(\d+(\.\d{3})?)(?=\s)', re.I)

jumlahnya = jumlah\_regex.search(kal[i])

if (jumlahnya != None):

min = span[i] - jumlahnya.span()[1]

temp = jumlahnya.group()

for match in jumlah\_regex.finditer(kal[i]):

if(min >= (span[i] - match.span()[1])):

min = span[i] - match.span()[1]

temp = match

jumlah[i] = temp.group()

return jumlah

def searchSpanPattern(kalimat, pattern):

pat = re.compile(r'(' + pattern + r')', re.I)

spanPat = []

for i in range(len(kalimat)):

posn = pat.search(kalimat[i])

for match in pat.finditer(kalimat[i]):

spanPat.append(match.span()[1])

return spanPat

def waktuRegex(kal, waktuArtikel):

x = len(kal)

waktu = [0 for waktu in range(x)]

for i in range(x):

#Asumsi waktu/tanggalnya sesuai seperti contoh text

#dimana ada 'Hari' lalu tanggal bebas, lalu ada "Jam" + WIB/WITA/WIT (asumsi juga waktu di Indonesia)

waktu\_regex = re.compile(r'((Senin|Selasa|Rabu|Kamis|Jumat|Sabtu|Minggu)?[\,\s]?\(?\d+[\/\-\s](\d+|Januari|Jan|Februari|Feb|Maret|Mar|April|Apr|Mei|Juni|Jun|Juli|Jul|Agustus|Agus|Agu|Ags|September|Sep|Oktober|Okt|November|Nov|Desember|Des)[\/\-\s]\d+\)?([,]?[ ]?Pukul|pukul)?(\s[\d]{1,2}[.:][\d]{2}\sWIB|WITA|WIT?)?)', re.I)

waktunya = waktu\_regex.search(kal[i])

if (waktunya == None):

waktu[i] = waktuArtikel

else:

waktu[i] = waktunya.group()

return waktu

def kalimatRegex(text, pattern):

kalimat\_regex = re.compile(r'(?:^|[\.\n]\s)(.\*?' + pattern + r'.\*?)(?=\.\s|\n)', re.I)

kal = kalimat\_regex.findall(text)

return kal

def getWaktuArtikel(text):

waktu\_artikel = re.compile(r'((Senin|Selasa|Rabu|Kamis|Jumat|Sabtu|Minggu)?[\,\s]?\s?\(?\d+[\/\-\s](\d+|Januari|Jan|Februari|Feb|Maret|Mar|April|Apr|Mei|Juni|Jun|Juli|Jul|Agustus|Agus|Agu|Ags|September|Sep|Oktober|Okt|November|Nov|Desember|Des)[\/\-\s]\d+\)?([,]?[ ]?Pukul|pukul)?(\s[\d]{1,2}[.:][\d]{2}\sWIB|WITA|WIT?)?)', re.I)

waktu = waktu\_artikel.search(text)

return waktu.group()

def mainRegex(text, keyword, filename, hasil2):

waktuArtikel = getWaktuArtikel(text)

low = text.lower()

pattern = keyword.lower()

posn = searchSpanPattern(low, pattern)

if (len(posn) < 0):

print("Keyword not found")

else:

kalimat = kalimatRegex(text, keyword)

span = searchSpanPattern(kalimat, keyword)

waktu = waktuRegex(kalimat, waktuArtikel)

jumlah = jumlahRegex(kalimat, keyword, span)

for i in range(len(kalimat)):

data = {}

data['jumlah'] = jumlah[i]

data['waktu'] = waktu[i]

data['kalimat'] = kalimat[i]

data['filename'] = filename

hasil2.append(data)

def openApp(upDir, pattern, opsi):

files = [f for f in listdir(upDir) if isfile(join(upDir, f))]

artikel = []

for op in files:

upPath = upDir + '/' + op

f = open(upPath, "r")

artikel\_2 = f.read()

artikel.append({"artikel": artikel\_2, "filename": op})

hasil = []

for berita in artikel:

text = berita["artikel"]

filename = berita["filename"]

if (opsi == "opsi1"):

mainKMP(text, pattern, filename, hasil)

elif (opsi == "opsi2"):

mainBM(text, pattern, filename, hasil)

elif (opsi == "opsi3"):

mainRegex(text, pattern, filename, hasil)

return hasil

* web.py

gui web dengan flask

import os

import sys

from os import listdir

from os.path import isfile, join

from flask import Flask, render\_template, request, redirect, url\_for, send\_from\_directory

from werkzeug.utils import secure\_filename

from main import \*

upDir = os.getcwd() + '/upload'

app = Flask(\_\_name\_\_)

app.config['UPLOAD\_FOLDER'] = upDir

app.config['ALLOWED\_EXTENSIONS'] = set(['txt'])

def openFile(filename):

return '.' in filename and \

filename.rsplit('.', 1)[1] in app.config['ALLOWED\_EXTENSIONS']

@app.route('/')

def index():

return render\_template('index.html')

@app.route('/upload', methods=['POST'])

def upload():

uploaded\_files = request.files.getlist("file[]")

nama\_file = []

for file in uploaded\_files:

if file and openFile(file.filename):

filename = secure\_filename(file.filename)

file.save(os.path.join(app.config['UPLOAD\_FOLDER'], filename))

nama\_file.append(filename)

return render\_template('upload.html', nama\_file=nama\_file)

@app.route('/upload/<filename>')

def uploaded\_file(filename):

return send\_from\_directory(app.config['UPLOAD\_FOLDER'],filename)

@app.route('/perihal')

def perihal():

return render\_template('perihal.html')

@app.route('/hasil', methods=['POST'])

def hasil():

pattern = request.form['pattern']

opsi = request.form['inlineRadioOptions']

send = openApp(upDir, pattern, opsi)

package = [send, pattern]

return render\_template('hasil.html', send=package)

if \_\_name\_\_ == '\_\_main\_\_':

app.run()

* index.html

<!DOCTYPE html>

<html lang="en">

<head>

</head>

<body>

<div class="container">

<div class="header">

<h3 class="text-muted"><center>"Ekstraksi Informasi Artikel"</center></h3>

</div>

<hr/>

<div>

<form action="upload" method="post" enctype="multipart/form-data">

<center><input type="file" multiple="" name="file[]" class="span3" /><br /></center>

<center><input type="submit" value="Upload" class="span2"></center>

</form>

</div>

</div>

</body>

</html>

* upload.html

<!DOCTYPE html>

<html lang="en">

<head>

</head>

<body>

<div class="container">

<div class="header">

<h3 class="text-muted"><center>"Ekstraksi Informasi Artikel"</center></h3>

</div>

<hr/>

<div class="header">

<h3 class="text-muted">File Berhasil di Upload</h3>

</div>

<form method="POST" action="hasil">

<span><center>Metode: </center></span>

<div>

<center><input type="radio" name="inlineRadioOptions" id="inlineRadio1" value="opsi1">

<label for="inlineRadio1"><span>Knuth-Morris-Pratt(KMP)</span> </label></center>

</div>

<div>

<center><input type="radio" name="inlineRadioOptions" id="inlineRadio2" value="opsi2">

<label for="inlineRadio2"><span>Boyer-Moore(BM)</span> </label></center>

</div>

<div>

<center><input type="radio" name="inlineRadioOptions" id="inlineRadio3" value="opsi3">

<label for="inlineRadio2"><span>Regular Expression(Regex)</span> </label></center>

</div>

<center><input name="pattern" placeholder="Masukan Keyword dan Tekan Enter"></center>

</form>

<hr/>

</div>

</div>

</body>

</html>

* hasil.html

<!DOCTYPE html>

<html lang="en">

<head>

<style>

</style>

</head>

<body>

<div class="container">

<div class="header">

<h3 class="text-muted"><center>"Ekstraksi Informasi Artikel"</center></h3>

</div>

<hr/>

<div>

<ul class="aturposisi">

</ul>

</div>

<form method="POST" action="hasil">

<span><center>Metode: </center></span>

<div>

<center><input type="radio" name="inlineRadioOptions" id="inlineRadio1" value="opsi1">

<label for="inlineRadio1"><span class="naikindikit">Knuth-Morris-Pratt(KMP)</span> </label></center>

</div>

<div>

<center><input type="radio" name="inlineRadioOptions" id="inlineRadio2" value="opsi2">

<label for="inlineRadio2"><span class="naikindikit">Boyer-Moore(BM)</span> </label></center>

</div>

<div>

<center><input type="radio" name="inlineRadioOptions" id="inlineRadio3" value="opsi3">

<label for="inlineRadio2"><span class="naikindikit">Regular Expression(Regex)</span> </label></center>

</div>

<center><input name="pattern" placeholder="Masukan Keyword dan Tekan Enter"></center>

</form>

</div>

<div>

<p><strong>Keyword</strong> : {{send[1]}}</p>

<br>

<p>Catatan: Jika ingin mengupload file kembali, silahkan kembali kehalaman upload file </p>

<p><strong>Hasil Ekstrasi Informasi: </strong></p>

{% for article in send[0]: %}

<div>

<p><strong>Jumlah </strong>: {{article['jumlah']}}</p>

<p><strong>Waktu </strong>: {{article['waktu']}}</p>

<p><strong>Kalimat </strong>: </p>

<p>{{article['kalimat']+' ('+article['filename']+')'}}</p>

</div>

{% endfor %}

<hr/>

<p><a href="{{ url\_for('perihal') }}">Perihal</a></p>

</div>

</div>

</body>

</html>

* perihal.html

<!DOCTYPE html>

<html lang="en">

<head>

<style>

.justify {

text-align: justify;

}

</style>

</head>

<body>

<h3 class="text-muted"><center>"Ekstraksi Informasi Artikel"</center></h3>

<h3 class="justify">Tentang Program dan Pembuat</h3>

<hr/>

<p class="justify">Pembuat Program: </p>

<p class="justify">Nama : Fikra Hadi Ramadhan</p>

<p class="justify">NIM : 13518036</p>

<p class="justify">Hal : Tugas Kecil 4 - Strategi Algoritma</p>

<p class="justify">Judul : Ekstraksi Informasi dari Artikel Berita dengan Algoritma Pencocokan String</p>

<br>

<p class="justify">Program Ekstraksi Informasi Artikel adalah sebuah program dimana pengguna bisa menggunakannya untuk mencari keyword pada banyak artikel sekaligus. Pengguna akan diminta untuk meng-upload beberapa file yang ingin digunakan, lalu memasukkan keyword yang ingin dicari pada file-file yang sudah di upload tersebut. Pengguna juga bisa memilih akan menggunakan algoritma KMP (Knuth-Morris-Pratt), BM (Boyer-Moore), atau Regex (Regular Expression). Jika algoritmanya benar, maka hasil yang dikeluarkan akan sama. Hasil yang ditampilkan adalah Jumlah(angka), Waktu, dan pada kalimat mana keywordnya ditemukan.</p>

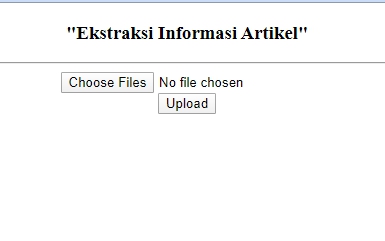
<p class="justify">Program ini memudahkan pengguna untuk mencari keyword yang diinginkan, tidak perlu memeriksa satu persatu ke artikel yang dimiliki.</p>

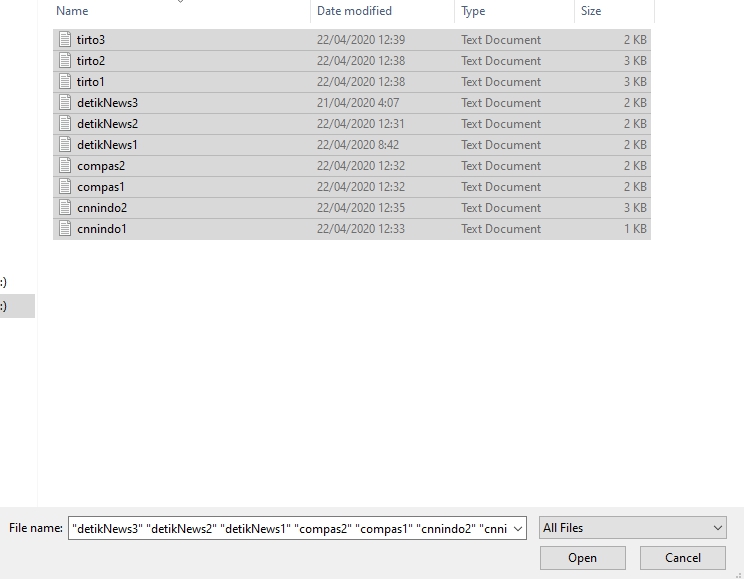
</body>

</html>

# BAB 3 SCREENSHOT INPUT-OUTPUT PROGRAM

3.1. Input File (index.html)



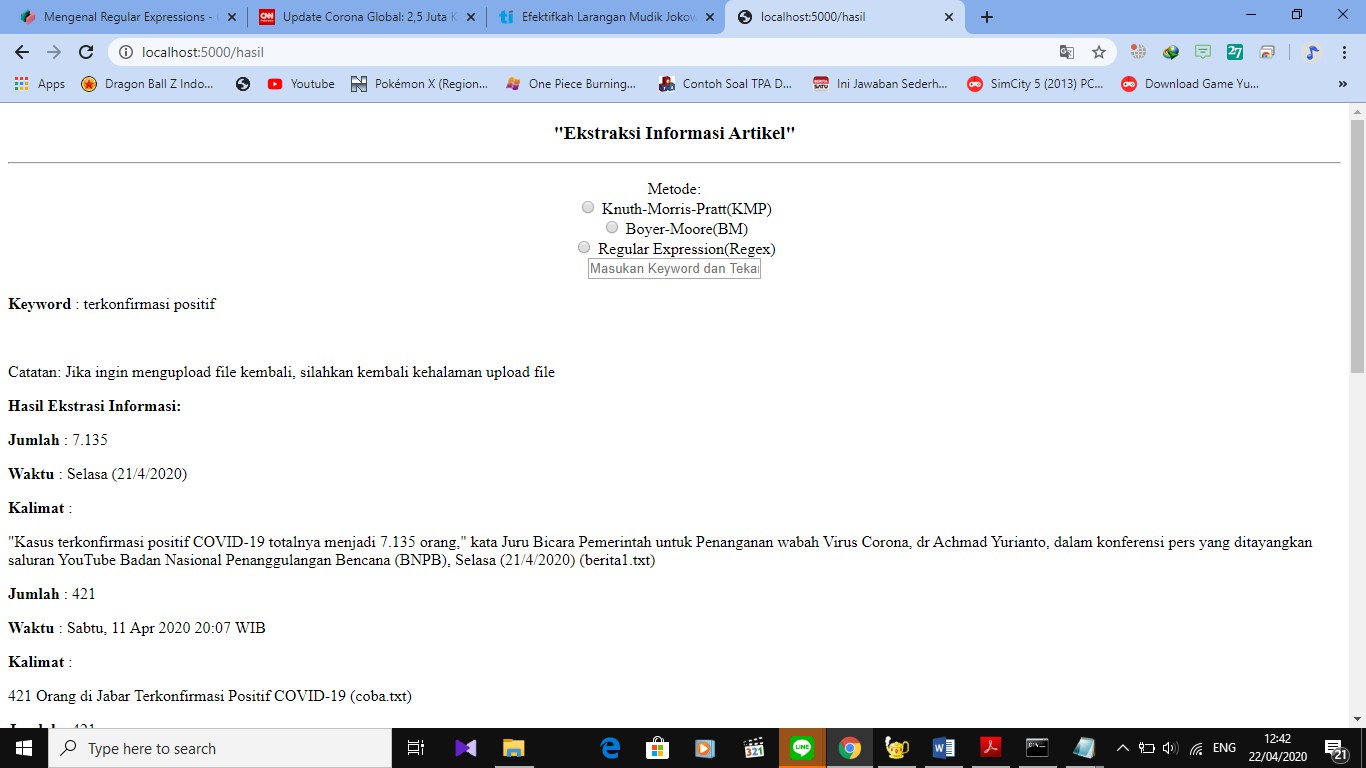


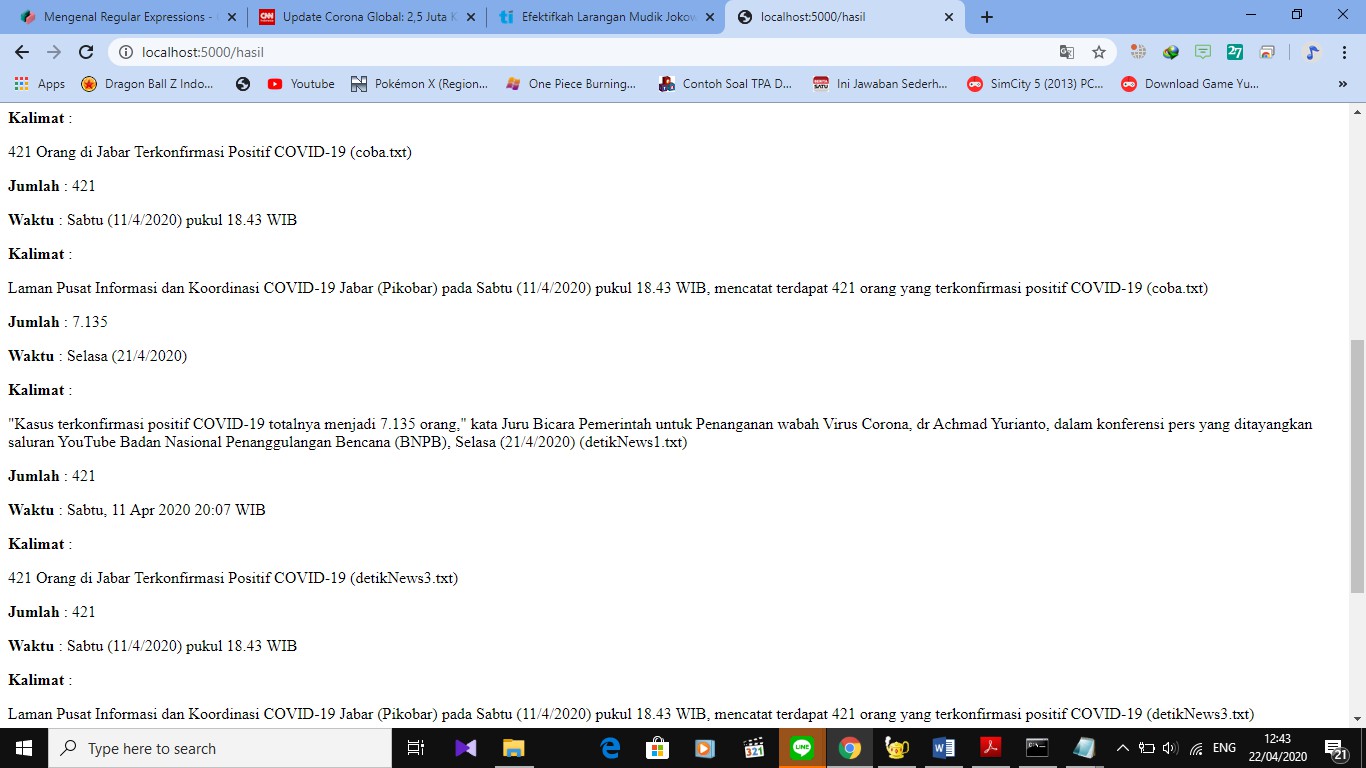


3.2. Input Keyword (upload.html)



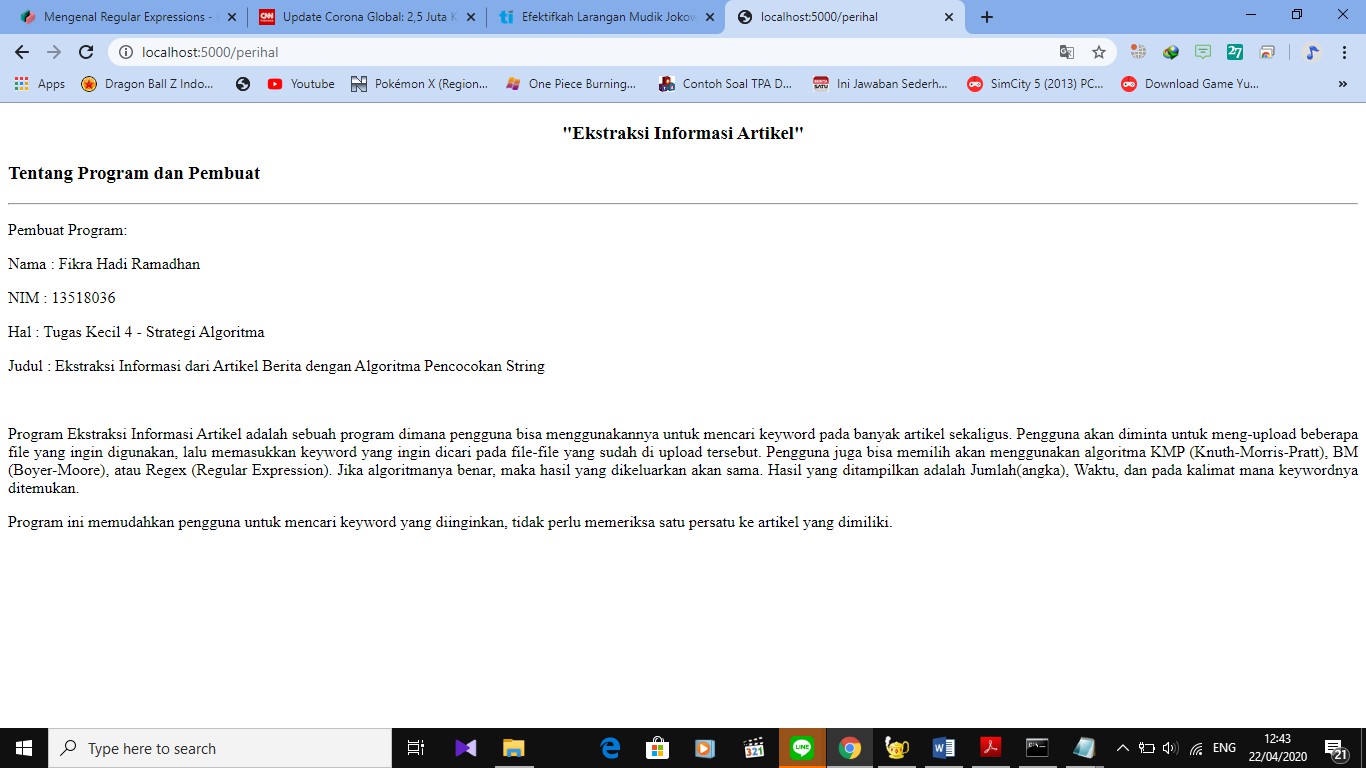
3.3. Output (hasil.html)





* 1. Perihal (perihal.html)





# DAFTAR PUSTAKA

<http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2019-2020/stima19-20.htm>

<https://gist.github.com/liulixiang1988/cc3093b2d8cced6dcf38>

<https://www.researchgate.net/publication/318670404_IMPLEMENTASI_ALGORITMA_KNUTH_MORRIS_PRATT_PADA_APLIKASI_PENERJEMAHAN_BAHASA_MANDAILING-INDONESIA>

<http://technoprast.blogspot.com/2011/04/kasus-pada-algoritma-boyer-moore.html>

<https://www.codepolitan.com/mengenal-regular-expressions-5a15003d9410a>

<https://m.detik.com/news/berita-jawa-barat/d-4985759/jabar-hari-ini-keluarga-perawat-dibacok-dan-pasien-positif-corona-pertama-di-tasik>

<https://nasional.kompas.com/read/2020/04/22/10230541/muhammadiyah-mulai-ramadhan-jumat-ini-berikut-imbauan-terkait-covid-19>

<https://www.google.com/amp/s/amp.kompas.com/tren/read/2020/04/22/082509265/update-virus-corona-di-dunia-22-april-25-juta-orang-terinfeksi-688639>

<https://m.cnnindonesia.com/nasional/20200421120855-20-495594/update-corona-21-april-7135-positif-616-meninggal>

<https://m.cnnindonesia.com/internasional/20200422071126-134-495898/update-corona-global-25-juta-kasus-688-ribu-sembuh>

<https://www.google.com/amp/s/amp.tirto.id/update-corona-indonesia-21-april-2020-data-covid-19-dunia-terkini-ePYZ>

<https://tirto.id/efektifkah-larangan-mudik-jokowi-untuk-cegah-penularan-covid-19-ePY9?utm_source=Tirtoid&utm_medium=Popular>

<https://tirto.id/mengapa-pemerintah-perlu-ungkap-angka-kematian-riil-terkait-corona-eP1d?utm_source=Tirtoid&utm_medium=Popular>

PENILAIAN

