Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**71230982**
Nama Lengkap	**Rainie Fanita Chrisabel Hadisantoso**
Minggu ke / Materi	**14 / Regular Expression**

SAYA MENYATAKAN BAHWA LAPORAN PRAKTIKUM INI SAYA BUAT DENGAN USAHA SENDIRI TANPA MENGGUNAKAN BANTUAN ORANG LAIN. SEMUA MATERI YANG SAYA AMBIL DARI SUMBER LAIN SUDAH SAYA CANTUMKAN SUMBERNYA DAN TELAH SAYA TULIS ULANG DENGAN BAHASA SAYA SENDIRI.

SAYA SANGGUP MENERIMA SANKSI JIKA MELAKUKAN KEGIATAN PLAGIASI, TERMASUK SANKSI TIDAK LULUS MATA KULIAH INI.

PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS KRISTEN DUTA WACANA
YOGYAKARTA
2024

MATERI : Regular Expression

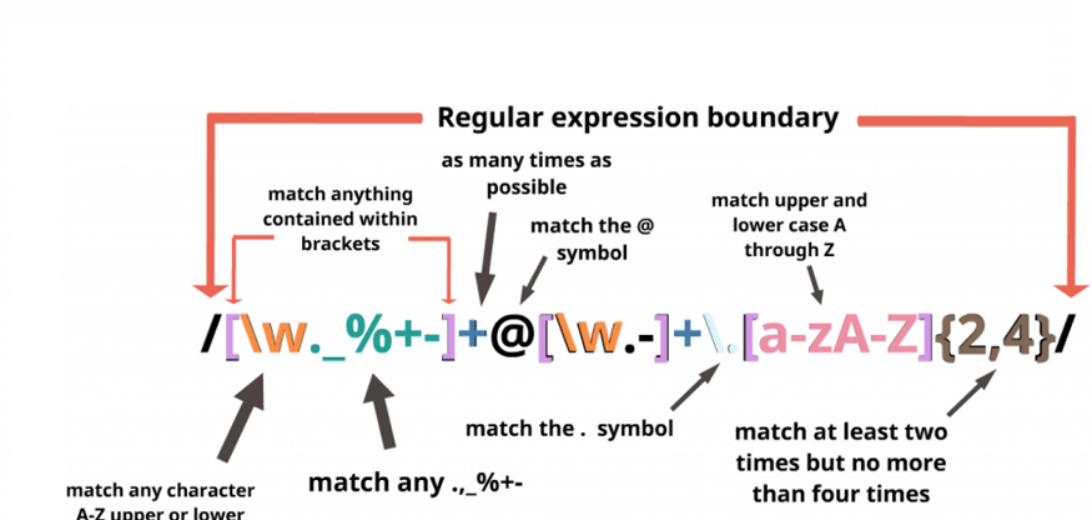
Dalam pertemuan ke 15 ini, disampaikan materi mengeak regular expression. Dari pertemuan ini diharapkan dapat dipelajari beberapa hal sebagai tujuan yang perlu dicapai:

1. Menjelaskan mengenai regular expression.
2. Menggunakan simbol dan fungsi regex secara umum.
3. Menggunakan library regex dari Python.
4. Memecahkan kasus-kasus regex dengan Python.

Pengantar Regex

Dari pelajaran string, kita sudah mempelajari cara-cara mengakses dan memanipulasi string, baik dari input user maupun file. Dapat disimpulkan bahwa pengolahan string cukup menyulitkan karena teknik-teknik pengolahan biasa yang terbatas. Namun sebetulnya ada teknik pengolahan lain yang lebih mudah dan bervariasi dengan bantuan modul tambahan dari luar modul bawaan Python, yaitu regular expression.

Regular expression adalah sebuah rangkaian pola dari karakter-karakter untuk mendapatkan makna yang sama dengan pola yang ingin ditemukan dalam string yang diolah. Regular expression dapat membantu kita dalam mencari string dengan pola tertentu, mengganti string dengan pola tertentu dan menhapus string dengan pola tertentu. Regular expression sebetulnya seperti parsing string yang telah kita pelajari seperti split() dan find(). Perbedaannya pola yang dapat dicari menggunakan regular expression bisa lebih kompleks sehingga sangat membantu pengolahan string yang lebih kompleks juga.



Regular expression sangatlah bebas dan bervariasi dalam mencari dan memanggil pola, namun regex memiliki ketentuan kode-kode sendiri yang cukup rumit. Karena itu tidak semua bahasa pemrograman mendukung penggunaan modul regular expression. Namun regular expression dapat digunakan dalam Python. Untuk memasukkan modul regular expression dalam kode Python kita, perlu digunakan perintah import re diawal kode.

```
import re

handle=open('mbox-short.txt')
```

```
count = 0
for line in handle:
    line=line.rstrip()
    if re.search('From:', line):
        count += 1
        print(line)
    print("Count: ",count)
```

Literal Character

Literal character adalah bentuk regular expression yang paling dasar untuk digunakan. Seperti dengan sebutannya, kita menggunakan karakter asli atau karakter aktual yang kita maksudakn secara langsung. Jadi jika kita ingin mencari dari kalimat “Hello, world!” misalnya, kita dapat langsung menggunakan kata yang mau kita cari. Misalnya yang dicari “world” maka ditulis saja “world” dan akan diproses kata tersebut sesuai dengan fungsi regular expression yang digunakan.

Meta Character

Karakter-karakter spesial yang memiliki makna tertentu. Jika dikombinasilkan dengan karakter literal maupun escaped karakter akan membangun sebuah kode dengan makna proses yang lebih kompleks namun dalam bentuk simpel. Dengan adanya meta karakter, ada begitu banyak variasi regular expression yang dapat digunakan untuk mencapai tujuan yang sama.

Karakter	Kegunaan	Contoh	Arti Contoh
[]	Kumpulan karakter	"[a-zA-Z]"	1 karakter antara a-z kecil atau A-Z besar
\{}	Karakter dengan arti khusus dan escaped character	\{}d	Angka / digit
.	Karakter apapun kecuali newline	say.n.	Tidak bisa diganti dengan karakter apapun, misal "sayang" akan valid
^	Diawali dengan	^From	Diawali dengan From
$	Dakhiri dengan	this$	Diakhiri dengan kata this
*	0 s/d tak terhingga karakter	\{}d*	ada digit minimal 0 maksimal tak terhingga
?	ada atau tidak (opsional)	\{}d?	Boleh ada atau tidak ada digit sebanyak
+	1 s/d tak terhingga karakter	\{}d+	Minimal 1 s/d tak terhingga karakter
{}	Tepat sebanyak yang ada para {}	\{}d{2}	Ada tepat 2 digit

()	Pengelompokan karakter / pola	(saya|kamu)	saya atau kamu sebagai satu kesatuan
|	atau	\{}d|\{}s	1 digit atau 1 spasi

Escaped Character

Karakter-karakter alfabet yang digunakan dalam regular expression memiliki maknanya sendiri, tidak sesuai karakter yang diberikan secara literal. Karena itu ada aturan-atura tertentu agar mengubah karakter yang digunakan sebagai meta karakter dan bukan karakter literal. Contohnya biasanya akan digunakan tanda garis miring "\" sebelum karakter.

Special Characters	Kegunaan	Contoh
\b	Digunakan untuk mengetahui apakah suatu pola berada di awal kata atau akhir kata	"R\bin" "Rain\b"
\d	Digunakan untuk mengetahui apakah karakter adalah sebuah digit (0 s/d 9)	\d
\D	Digunakan untuk mengetahui apakah karakter yang bukan digit	\D
\s	Digunakan untuk mengetahui apakah karakter adalah whitespace (spasi, tab, enter)	\s
\S	Digunakan untuk mengetahui apakah karakter adalah BUKAN whitespace (spasi, tab, enter)	\S
\w	Digunakan untuk mengetahui apakah karakter adalah word (a-z, A-Z, 0-9, dan _)	\w
\W	Digunakan untuk mengetahui apakah karakter adalah BUKAN word (a-z, A-Z, 0-9, dan _)	\W
\A	Digunakan untuk mengetahui apakah karakter adalah berada di bagian depan dari kalimat	"\AThe"
\Z	Digunakan untuk mengetahui apakah karakter adalah berada di bagian akhir dari kalimat	"End\Z"

Set of Character

Set of character atau disebut juga character class adalah list berisi karakter-karakter yang didefinisikan dengan penggunaan meta karakter "[]". Dapat diberikan lebih dari satu set of character dalam sebuah fungsi regular expression. Namun cara kerjanya tidak dibaca satu set dulu sampai habis tapi 1 karakter

dalam set pertama, pindah 1 karakter di set kedua, lalu balik ke set pertama untuk mengambul karakter kedua, dan seterusnya.

[abc]	Mencari pola 1 huruf a, atau b, atau c
[a-c]	Mencari pola 1 huruf a s/d c
[^bmx]	Mencari pola 1 huruf yang bukan b,m, atau x
[012]	Mencari pola 1 huruf 0, atau 1, atau 2
[0-3]	Mencari pola 1 huruf 0 s/d 3
[0-2][1-3]	Mencari pola 2 huruf: 01, 02, 03, 11, 12, 13, 21, 22, 23
[a-zA-Z]	Mencari pola 1 huruf a-Z

Fungsi Regex

Dalam materi regular expression ini, kita hanya akan menggunakan 4 fungsi regular expression saja, yaitu findall, search, split, sub. Semua variasi karakter meta dan escape untuk mendapat hasil yang kita inginkan kita masukkan kedalam fungsi yang diinginkan. Baru program dapat memproses regular expression yang ingin kita lakukan. Untuk menggunakan fungsi regular expression biasanya rumusnya adalah : re.fungsi(rumus regular expression).

Nama Fungsi	Kegunaan
findall	mengembalikan semua string yang sesuai pola (matches)
search	mengembalikan string yang sesuai pola (match)
split	memecah string sesuai pola
sub	mengganti string sesuai dengan pola yang cocok

Kegiatan Praktikum

1. Findall

   Findall diguanakan untuk mencari semua pola yang di berikan dalam string yang hendak diolah. Misalnya kita memasukkan pola “mata” untuk dicaru, maka program akan menampilkan semua kata yang mengandung “mata”.

```
import re
txt = "Sang mata-mata sedang memata-matai kasus kaca mata di toko
Matahari"
x = re.findall("mata", txt)
y = re.findall("saya", txt)
for i in x:
    print(i)

if (y):
```

```
    print("Ada yang cocok!")
else:
    print("Tidak ada yang cocok!")
```

2. Search

Search digunakan ntuk mencari apakah ada pola yang kita berikan dalam string yang hendak diolaj. Fungsi search hanya mencari dan memberi jawaban True atau False, tidak memanggil kata yang dicari.

```
import re
txt = "Sang mata-mata sedang memata-matai kasus kaca mata di toko
Matahari"
x = re.search("\s", txt)
y = re.search("saya", txt)

print("Spasi ditemukan di:", x.start())
print(y)
```

3. Split

Sama seperti fungsi split pada string biasa, fungsi reguler expression split ini digunakan untuk memisahkan string sesuai pola yang kita berikan. Misalnya spasi atau huruf atau titik.

```
import re
txt = "The rain in Spain"
x = re.split("\s", txt)
print(x)
y = re.split("\s", txt, 1) #split 1 kata pertama
print(x)
```

4. Sub

Sub digunakan untuk mengganti pola yang kita inginkan dengan kata lain pada sebuah string. Sistemnya adalah fungsi mencari pola yang kita inginkan, jika ada maka akan diganti dengan kata pengganti yang kita berikan. Dalam menggunakan sub kkta dapat memberikan batasan penggantian kata.

```
import re
txt = "Sang mata-mata sedang memata-matai kasus kaca mata di toko
Matahari"
x = re.sub("\s", "-", txt) #mengganti spasi dengan -
print(x)
y = re.sub("\s", "*", txt, 2) #mengganti spasi dengan * 2 saja
print(y)
```

BAGIAN 2: LATIHAN MANDIRI (60%)

SOAL 1

Tulis jawaban anda untuk soal nomor 1 di sini. Hapus paragraf ini.

a. Source code

```
import re
from datetime import datetime

rn = datetime.now()

yr = rn.year
mh = rn.month
dy = rn.day
tgl = (f'{dy}-{mh}-{yr}')
gp = re.compile(r'\d{4}-\d{2}-\d{2}')
text = open("tgl.txt", "r")
t = text.read()
cari = gp.findall(t)
x = []
for i in cari:
    a = i.split("-")
    b = (a[2]+'-'+a[1]+'-'+a[0])
    x.append(b)
form = '%d-%m-%Y'
tgl = datetime.strptime(tgl, form)
for j in x:
    c = datetime.strptime(j, form)
    d = tgl - c
    print(f'{tgl} 00:00:00 selisih {d.days} hari')
```

b. Output

```
PS C:\Users\asus\Downloads\File> python -u "c:\Users\asus\Downloads\File
2024-06-04 00:00:00 00:00:00 selisih 28781 hari
2024-06-04 00:00:00 00:00:00 selisih 87133 hari
2024-06-04 00:00:00 00:00:00 selisih 88020 hari
2024-06-04 00:00:00 00:00:00 selisih 49341 hari
```

c. Penjelasan

1. Mengimpor modul `re` untuk menggunakan regular expression dan modul `datetime` untuk bekerja dengan tanggal dan waktu.
2. Mendapatkan tanggal dan waktu saat ini menggunakan `datetime.now()` dan menyimpannya dalam variabel `rn`
3. Mendapatkan tahun (`yr`), bulan (`mh`), dan hari (`dy`) dari tanggal saat ini.

4. Membentuk string `tgl` yang berisi format tanggal yang diinginkan (`dd-mm-yyyy`) menggunakan f-string.
5. Membuat pola regex (`gp`) untuk mencocokkan string dengan format tanggal.
6. Membuka file "tgl.txt" dalam mode baca ("r") dan membaca isi teksnya ke dalam variabel `t`.
7. Menggunakan regex untuk mencocokkan semua tanggal dalam teks (`t`) dan menyimpannya dalam list `cari`.
8. Mengonversi format tanggal yang ditemukan dalam list `cari` dari `yyyy-mm-dd` menjadi `dd-mm-yyyy` dan menyimpannya dalam list `x`.
9. Mendefinisikan format tanggal yang diinginkan (`'%d-%m-%Y'`) dalam variabel `form`.
10. Mengonversi tanggal saat ini (`tgl`) ke dalam format yang telah ditentukan (`form`) menggunakan `datetime.strptime()`.
11. Melakukan iterasi pada setiap tanggal dalam list `x` untuk menghitung selisih hari dengan tanggal saat ini (`tgl`).
12. Menghitung selisih waktu antara tanggal saat ini (`tgl`) dengan tanggal dalam list `x` menggunakan operator `-`.
13. Mencetak hasil dalam format yang diinginkan, yaitu tanggal saat ini (`tgl`), waktu 00:00:00, dan selisih hari (`d.days`).

SOAL 2

Tulis jawaban anda untuk soal nomor 2 di sini. Format untuk soal nomor 3 dan seterusnya juga sama.

a. Source code

```
import re
import random
import string

g = re.compile(r'\S+@\S+')
text = open("akun.txt", "r")
t = text.read()
cari = g.findall(t)

for k in cari:
    at = k.split('@')
    h = at[0]
    pw_alfa = string.ascii_letters + string.digits
    pw = ''.join(random.choice(pw_alfa) for i in range(8))
    print(f'{k}: {h}, password: {pw}')
```

b. Output

```
PS C:\Users\asus\Downloads\File> python -u "c:\Users\asus\Downloads\
atihan14.2.py"
anton@mail.com: anton, password: jD08XAWS
budi@gmail.co.id: budi, password: 8BqVDLeW
slamet@getnada.com: slamet, password: hDKTxgsB
matahari@tokopedia.com: matahari, password: tfqVFJ8m
PS C:\Users\asus\Downloads\File> python -u "c:\Users\asus\Downloads\
atihan14.2.py"
anton@mail.com: anton, password: F2AGzaY6
budi@gmail.co.id: budi, password: 7GQprYWf
slamet@getnada.com: slamet, password: kLsEu0Po
matahari@tokopedia.com: matahari, password: z1ziSfEu
```

c. Penjelasan

1. Mengimpor modul `re` untuk menggunakan regular expression dan modul `random` dan `string` untuk menghasilkan kata sandi acak.
2. Membuat pola regex (`g`) untuk mencocokkan format alamat email.
3. Membuka file "akun.txt" dalam mode baca ("r") dan membaca isi teksnya ke dalam variabel `t`.
4. Menggunakan regex untuk mencocokkan semua alamat email dalam teks (`t`) dan menyimpannya dalam list `cari`.
5. Melakukan iterasi pada setiap alamat email dalam list `cari`.
6. Memisahkan alamat email menjadi bagian sebelum dan setelah "@" menggunakan method `split()`.
7. Menggunakan modul `string` untuk mendefinisikan karakter alfabet dan digit yang digunakan untuk membuat kata sandi acak.
8. Menggunakan modul `random` untuk memilih karakter secara acak dari kumpulan karakter alfabet dan digit yang telah ditentukan sebelumnya.
9. Menggabungkan karakter-karakter yang terpilih menjadi sebuah string menggunakan `join()` untuk membentuk kata sandi acak dengan panjang 8 karakter.
10. Mencetak alamat email (`k`), bagian sebelum "@" sebagai username (`h`), dan kata sandi acak yang telah dibuat (`pw`).