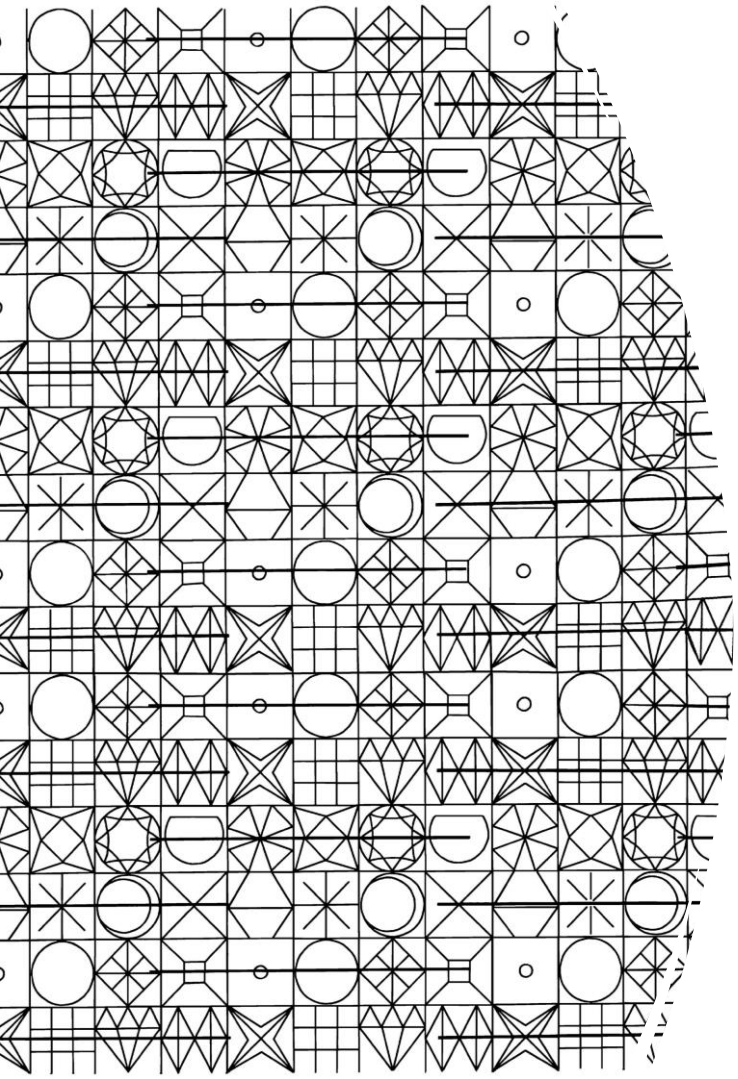
String Matching dengan Regular Expression

Masayu Leylia Khodra

Referensi:

Chapter 2 of *An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition,* by Daniel Jurafsky and James H. Martin

15-211 Fundamental Data Structures and Algorithms, by Ananda Gunawardena

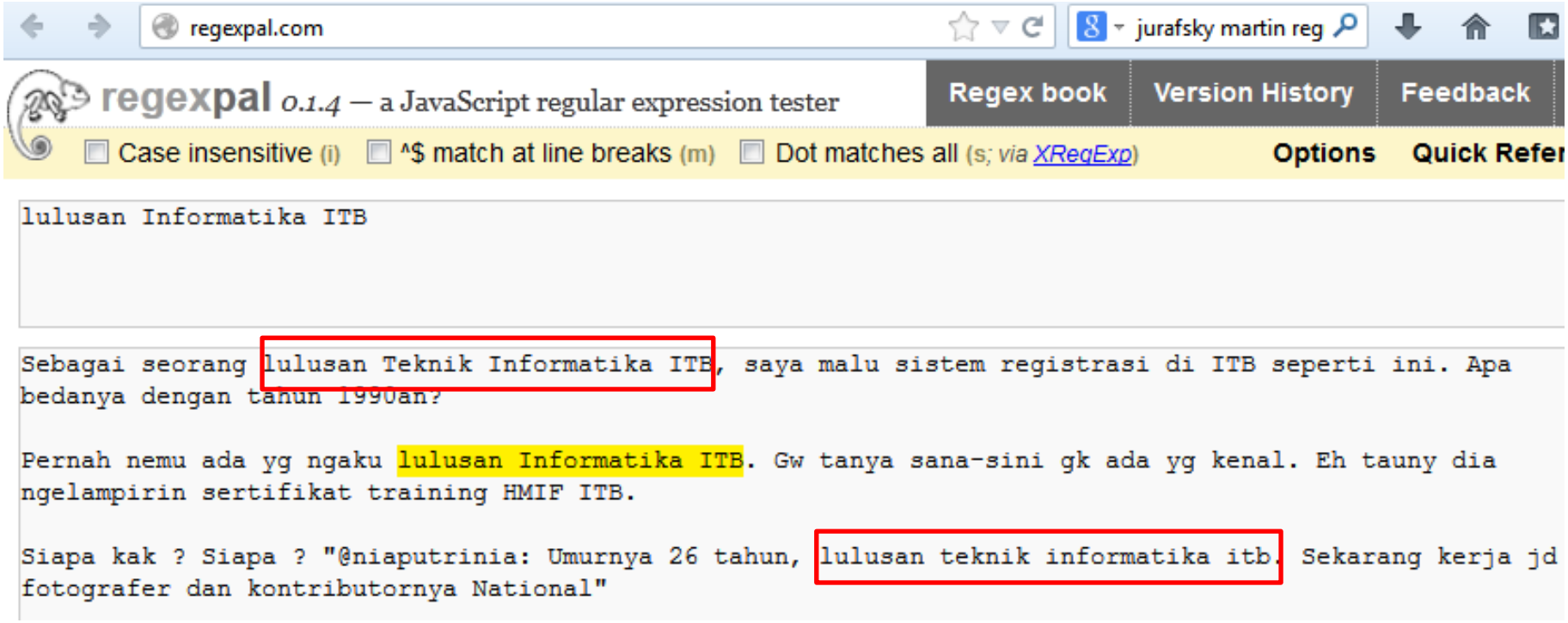


String Matching: Definisi

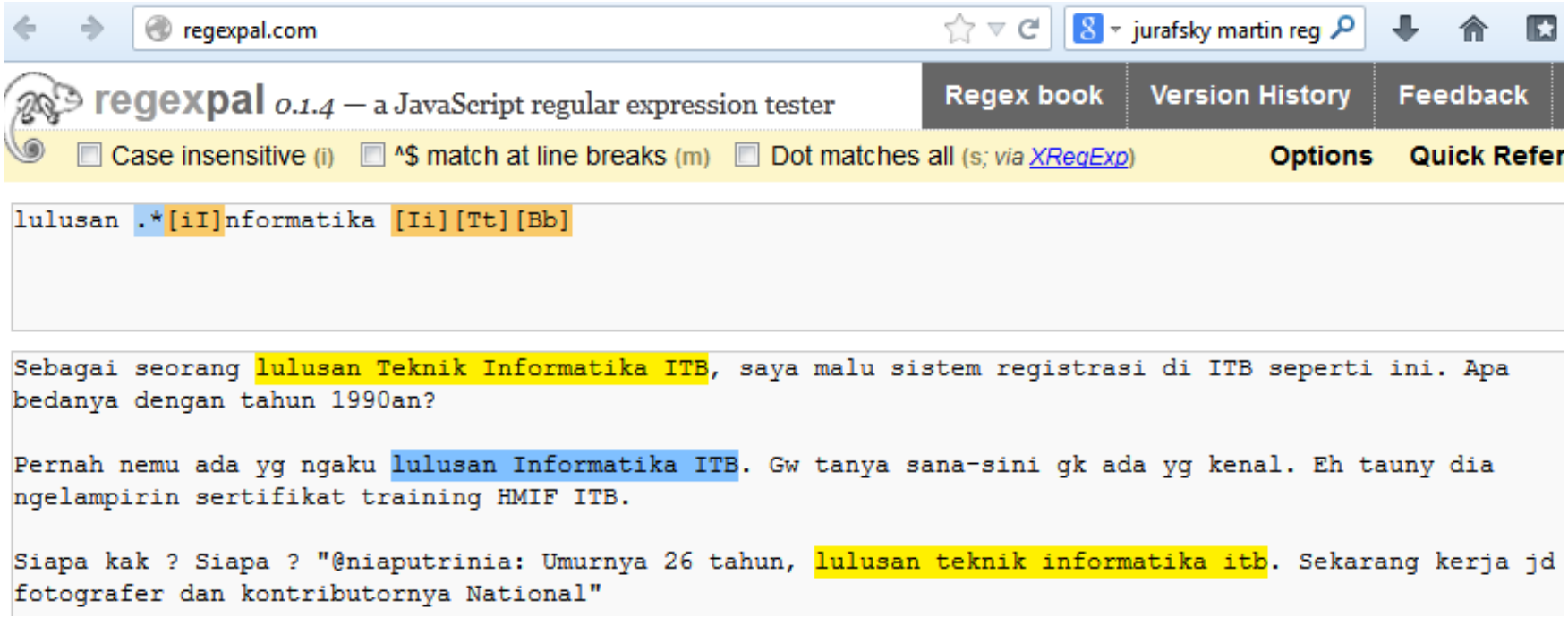
- Diberikan:
  1. *T*: teks (*text*), yaitu (*long*) *string* yang panjangnya *n* karakter
  2. *P: pattern*, yaitu *string* dengan panjang *m* karakter (asumsi *m* <<< *n*) yang akan dicari di dalam teks.

Carilah (*find* atau *locate*) di dalam teks yang bersesuaian dengan *pattern*.

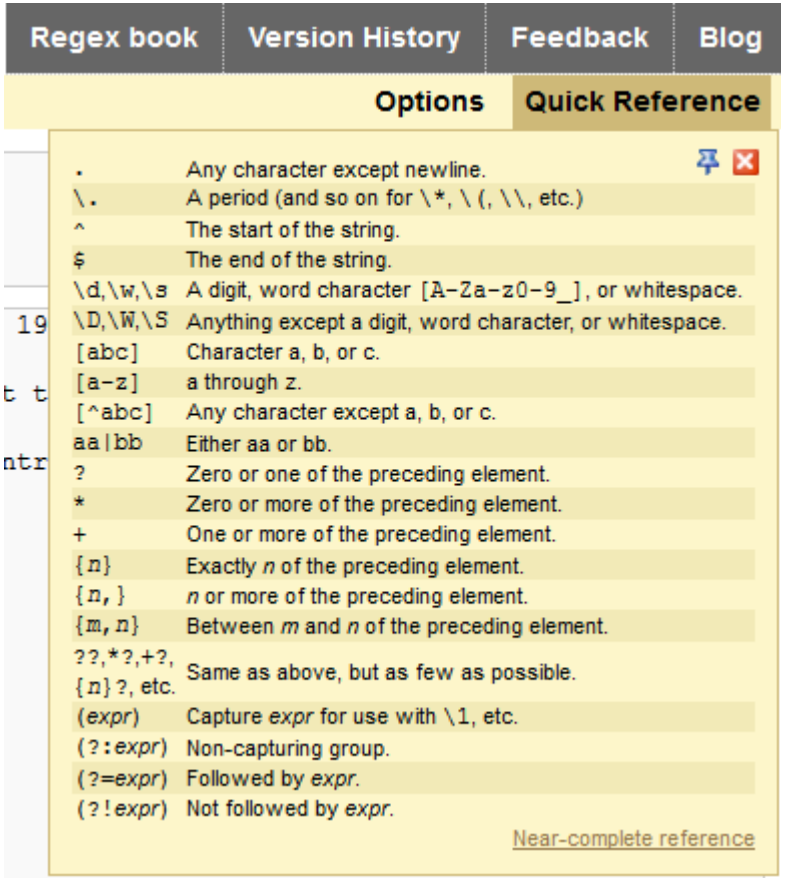
Contoh 1: Exact Matching

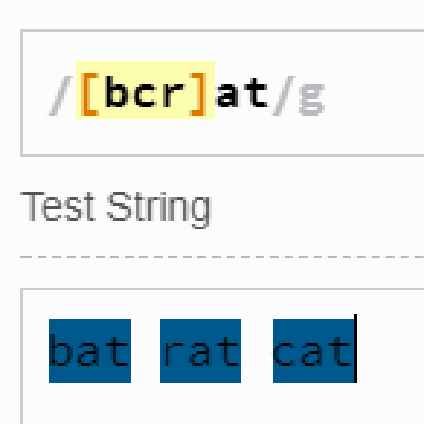


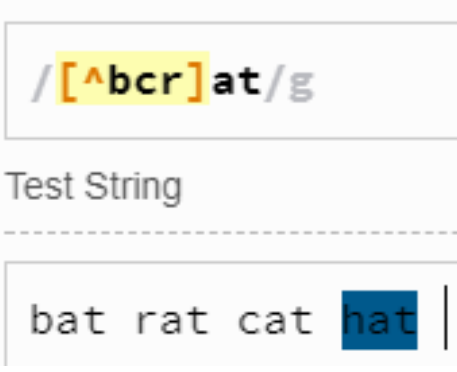
Contoh 2: Regex Matching

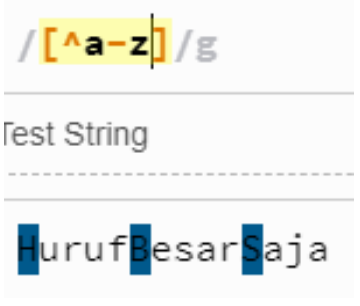


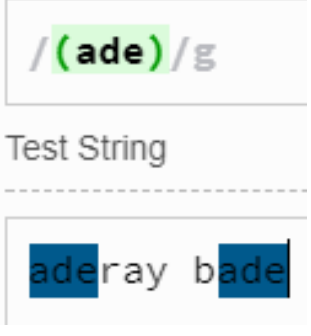
Notasi Umum Regex

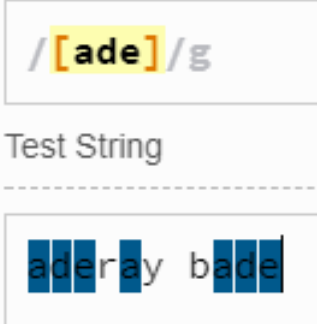




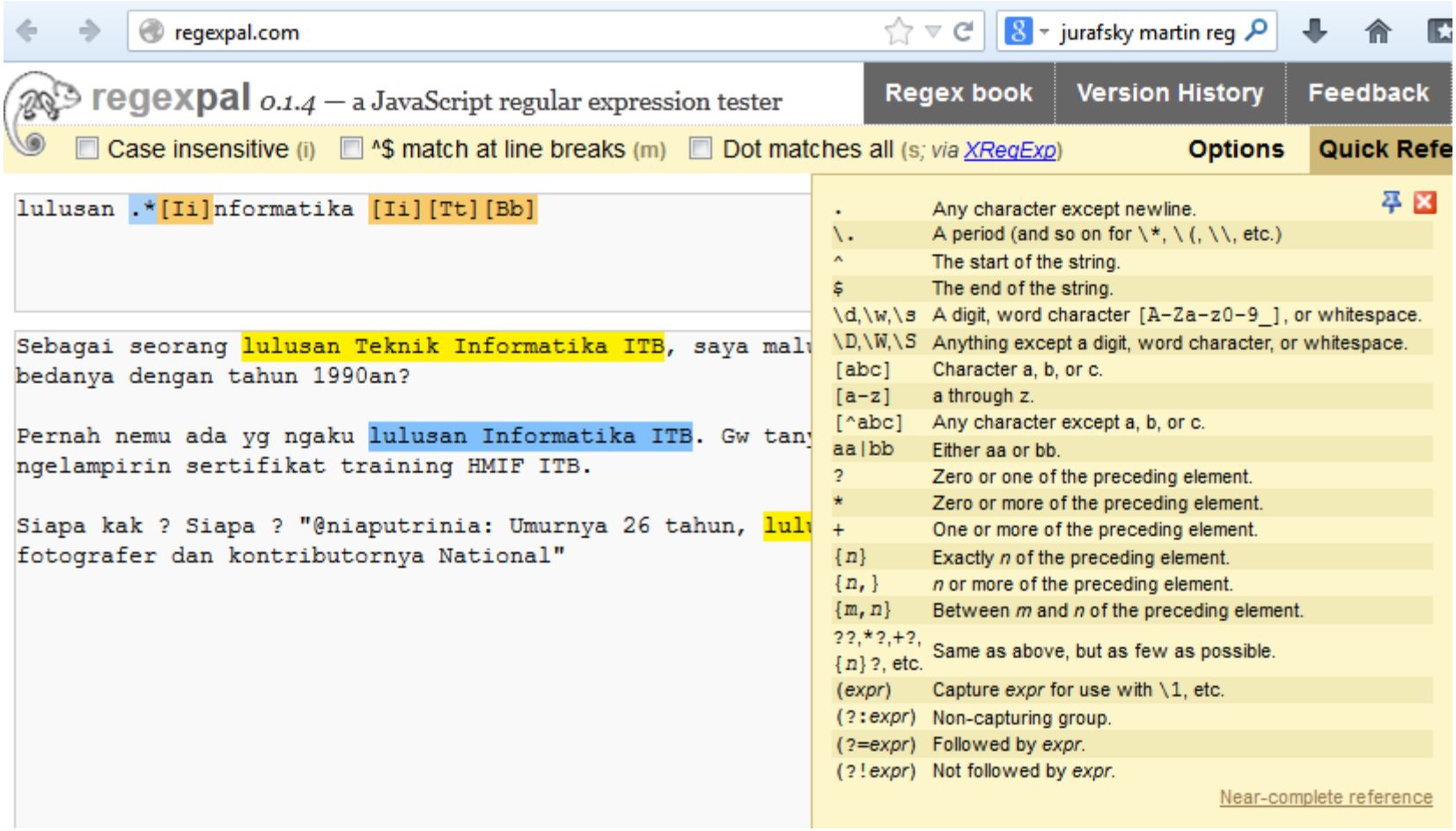








Contoh 2: Regex



Basic Regular Expression Patterns

brackets `[]`: `disjunction`

RE	Match	Example Patterns
`/[wW]oodchuck/`	Woodchuck or woodchuck	"Woodchuck"
`/[abc]/`	'a', 'b', *or* 'c'	"In uomini, in soldati"
`/[1234567890]/`	any digit	"plenty of 7 to 5"

Brackets `[]` ditambah garis sambung: range

RE	Match	Example Patterns Matched
`/[A-Z]/`	an uppercase letter	"we should call it 'Drenched Blossoms'"
`/[a-z]/`	a lowercase letter	"my beans were impatient to be hoed!"
`/[0-9]/`	a single digit	"Chapter 1: Down the Rabbit Hole"

Basic Regular Expression Patterns

- caret ^ : negasi

RE	Match (single characters)	Example Patterns Matched
[^A-Z]	not an uppercase letter	"Oyfn pripetchik"
[^Ss]	neither 'S' nor 's'	"I have no exquisite reason for't"
[^\.]	not a period	"our resident Djinn"
[e^]	either 'e' or '^'	"look up ^ now"
a^b	the pattern 'a^b'	"look up a^b now"

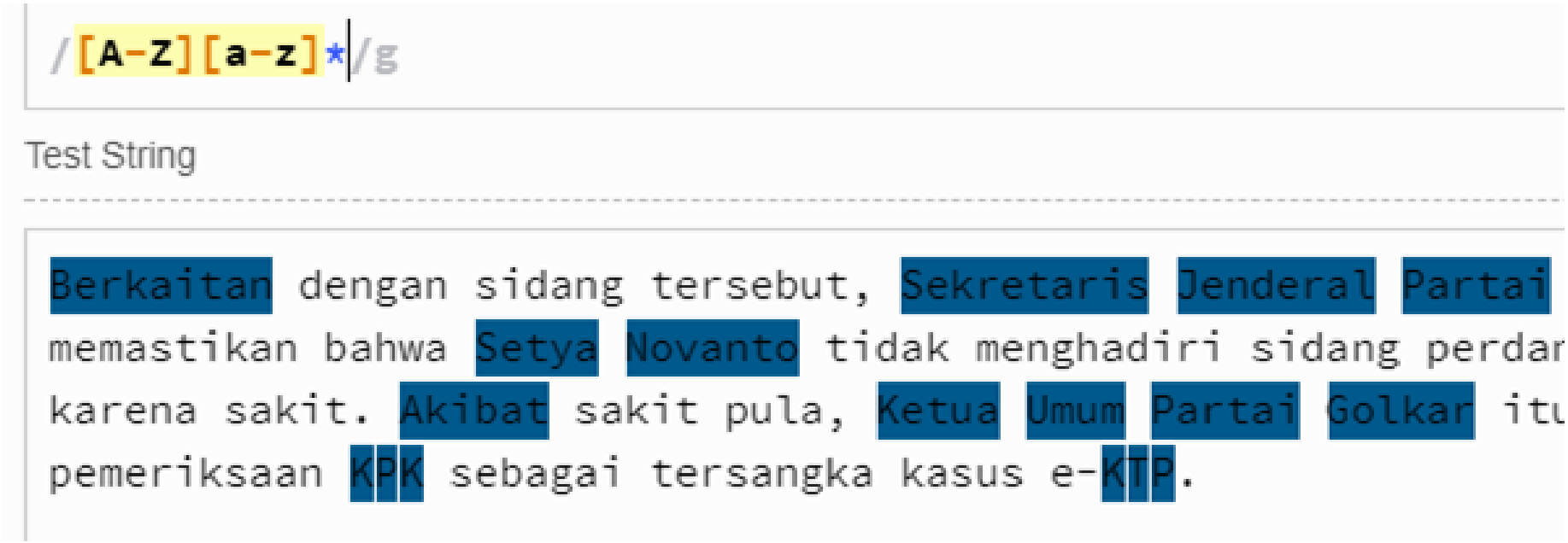
- Tanda tanya ? : bisa ada bisa tidak

RE	Match	Example Patterns Matched
woodchucks?	woodchuck or woodchucks	"woodchuck"
colou?r	color or colour	"colour"

- Titik: . any character

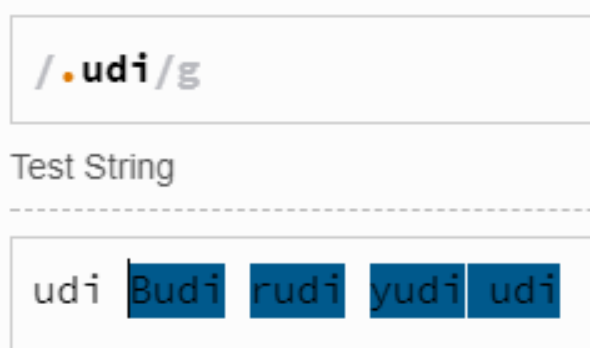
RE	Match	Example Patterns
/beg.n/	any character between *beg* and *n*	begin, beg'n, begun

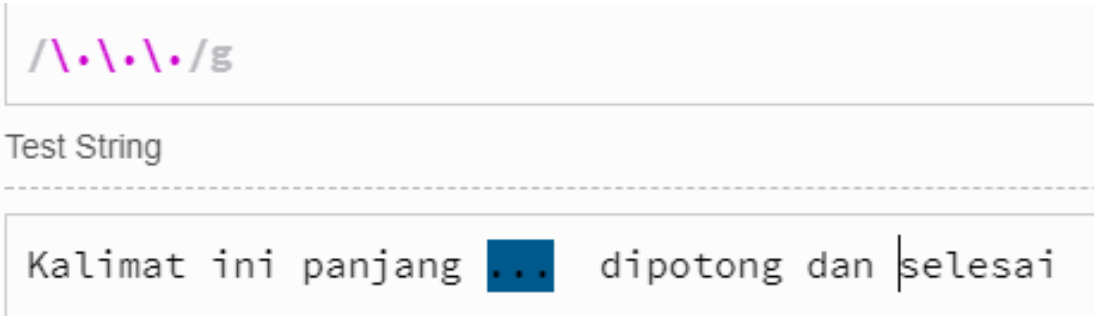
Regex Kata berawal Huruf Kapital



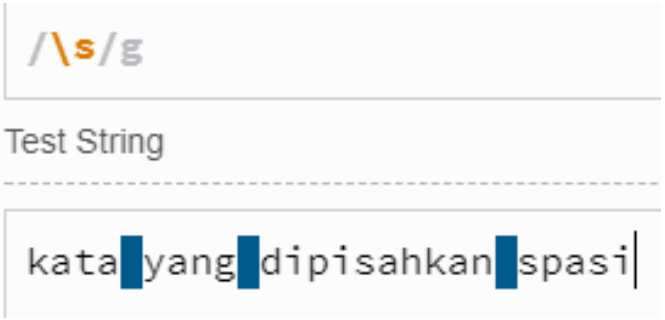
`[A-Z][a-z]*` : Alfabet huruf besar yang dilanjutkan dengan nol atau banyak huruf kecil

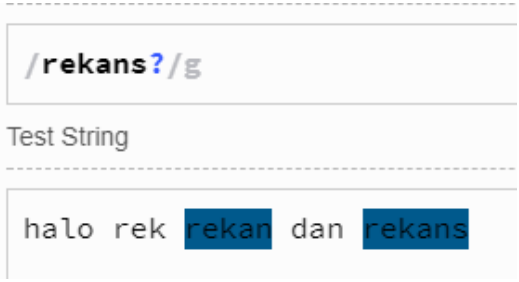
Notasi Regex: Contoh

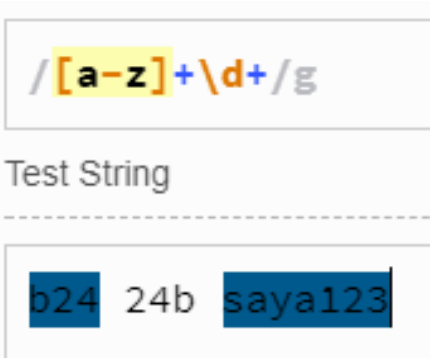




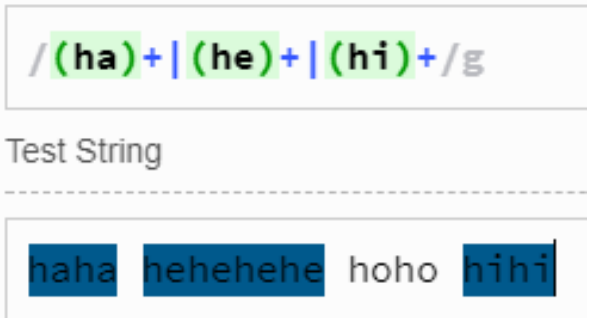
Metacharacter titik “.” menyatakan karakter apapun (kiri). Gunakanlah backslash ‘\’ untuk metacharacter.

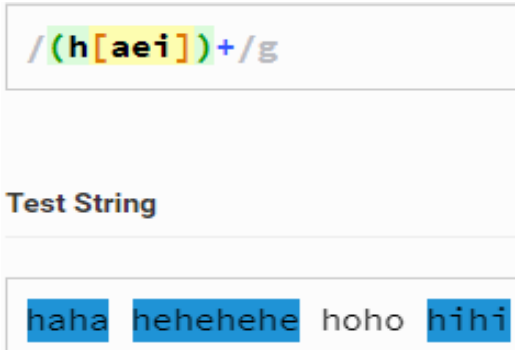


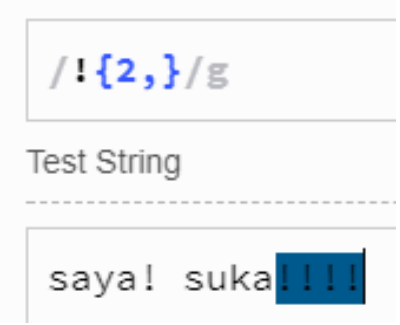




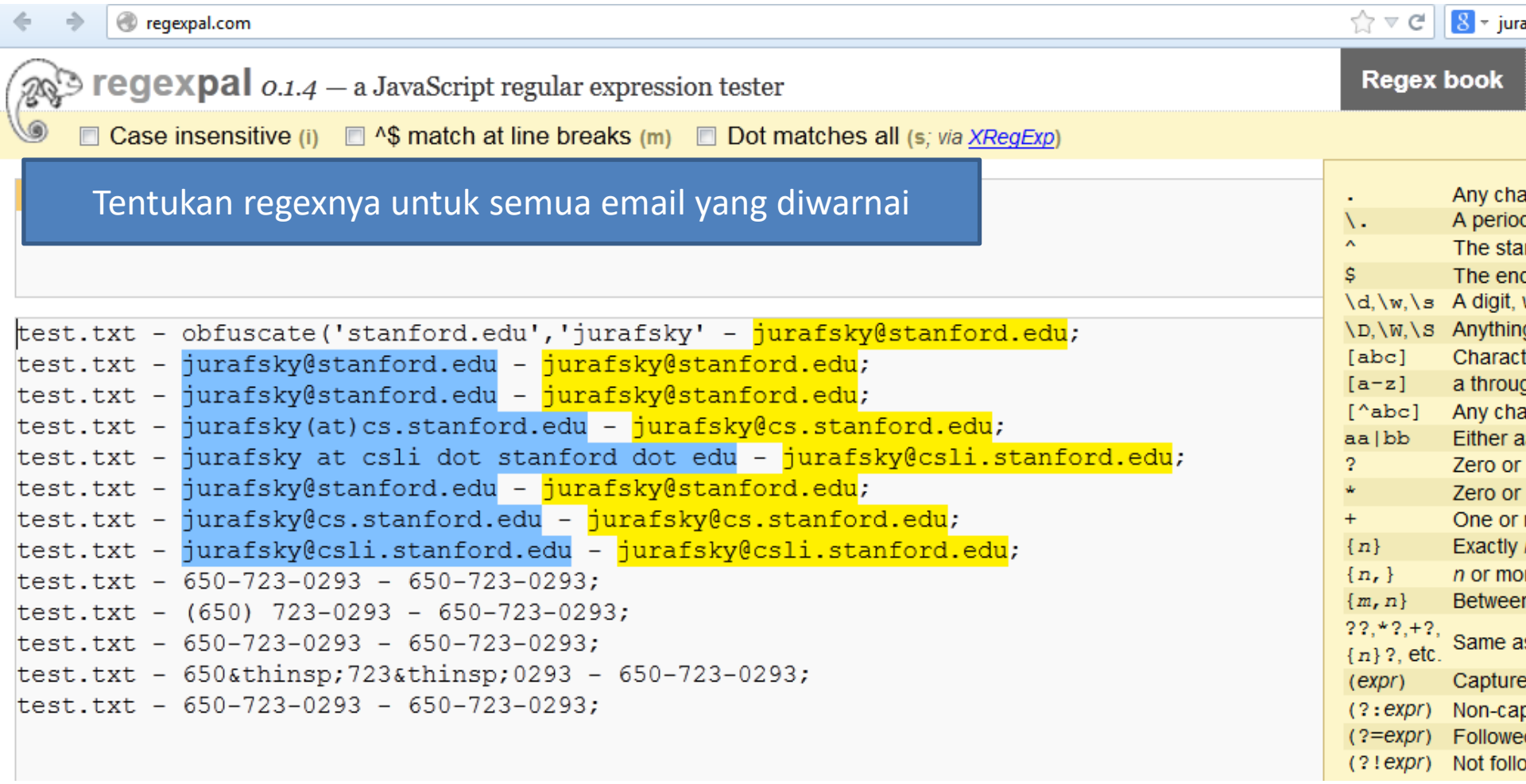
Notasi Regex: Contoh



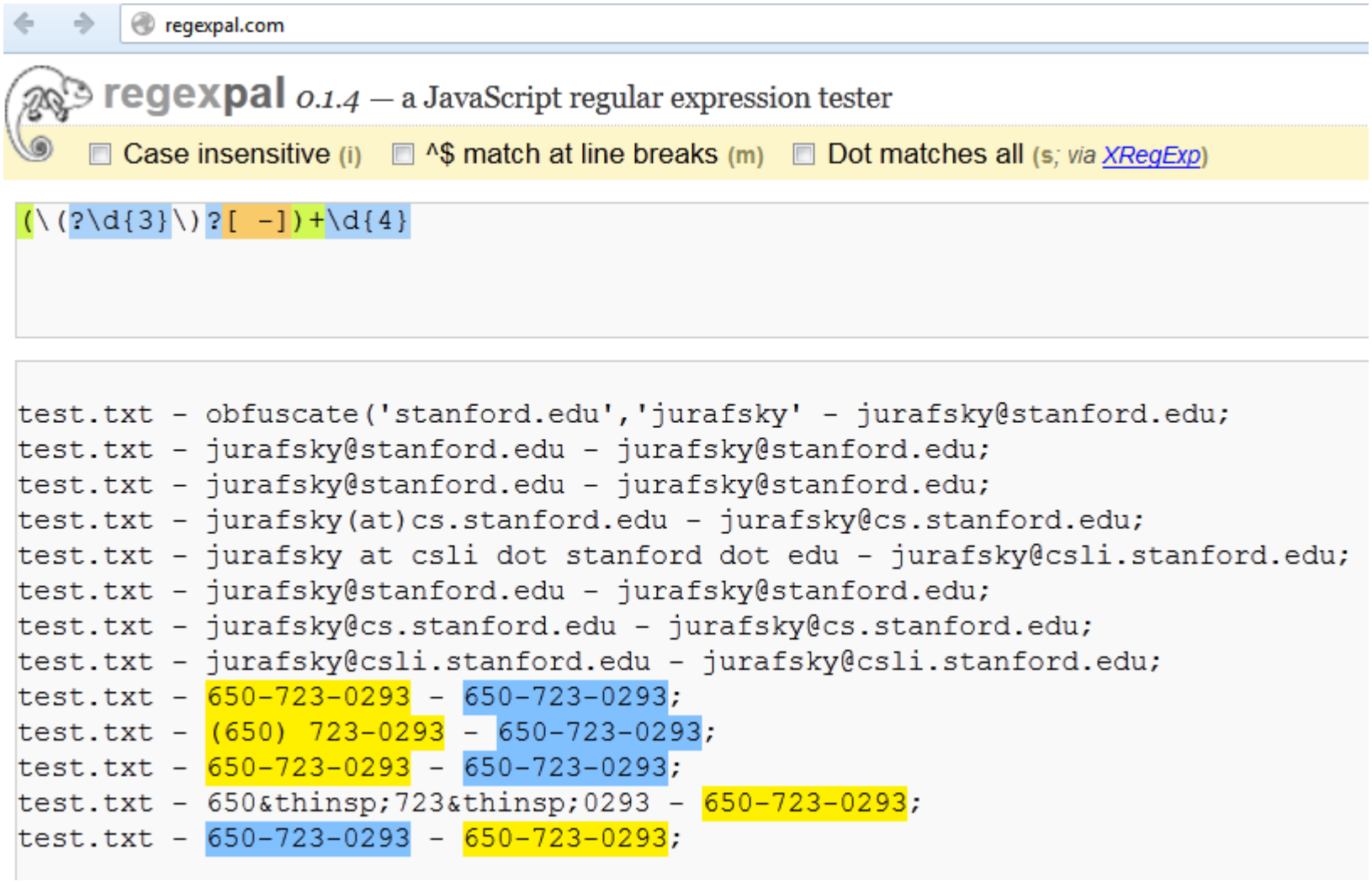




Contoh 3: Regex for Email



Contoh 4: Regex for Phone Number



Knowledge check Regex

Pelajarilah modul regex: https://docs.google.com/document/d/1ls6h1A6m-Zhzw6e5eriwMNUAG0D1iwL-eVmVMS2XQoc/edit?usp=sharing

Kerjakanlah Latihan 1-3 secara mandiri (tidak dikumpulkan).

Untuk Latihan 4, gunakanlah https://www.regexpal.com/ (tidak dikumpulkan)

Regex di Java

regexpal 0.1.4 — a JavaScript regular expression tester

Regex book

Case insensitive (i) ^$ match at line breaks (m) Dot matches all (s; via XRegExp)

```
\d{2}\.\d{2}
```

```
#lalinBDG 09.16 : yg mau ke jln sudirman dsk, jg

@infobdg: #lalinBDG 08.23 : Macet (lagi) rancaek

#suaraBDG via @dionmudjenan: Hatihati jembatan p

RT @quinsymegamira: leuwipanjang banyak anak jal
```

```
public static void extraction(Pattern myPattern,String str) {
    String extract;

    Matcher m;
    m = myPattern.matcher(str);
    while(m.find()) {
        extract = m.group();
        System.out.println(extract);
    }

}

public static void main(String[] args) {
    resources.extraction(Pattern.compile("\\d{2}\\.\\d{2}"), "@infobdg: #
```

Debugger Console × veritransTools (run) ×

```
run:
08.23
BUILD SUCCESSFUL (total time: 0 seconds)
```

Regex using Python

- re.**compile**(*pattern*, *flags=0*)
  Compile a regular expression pattern into a regular expression object, which can be used for matching using its match(), search() and other methods, described below.
- Pattern.**search**(*string*[, *pos*[, *endpos*]])
  Scan through *string* looking for the **first location** where this regular expression produces a match, and return a corresponding match object. Return None if no position in the string matches the pattern; note that this is different from finding a zero-length match at some point in the string.
  The optional second parameter *pos* gives an index in the string where the search is to start; it defaults to 0. The optional parameter *endpos* limits how far the string will be searched;

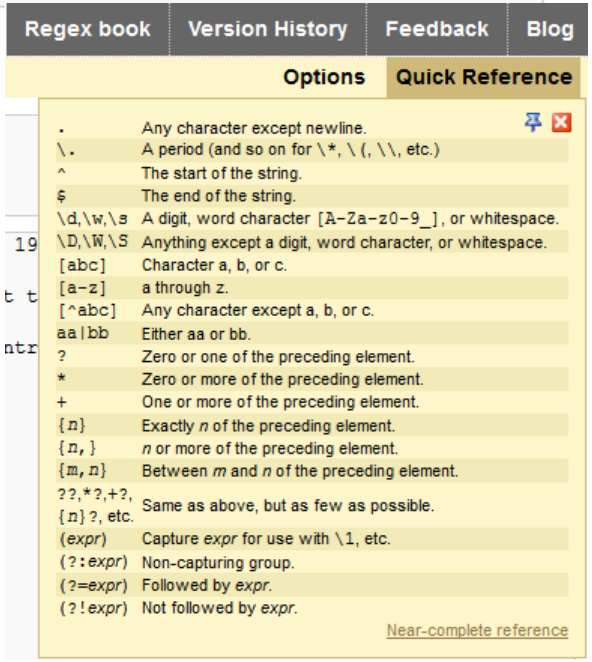
https://docs.python.org/3/library/re.html

Regex using Python

```python
import re

#Compile a regular expression pattern into a regular expression object
pattern = re.compile(r"(\d{4})")

#Scan through str looking for 1st loc where this regex produces a match,
#and return a corresponding match object.
pattern.search("17 Agustus 1945 - 2022")
```

```
<re.Match object; span=(11, 15), match='1945'>
```

\d{4}: digit characters exactly 4 characters



Regex using Python

- Pattern.**match**(*string*[, *pos*[, *endpos*]])
  If zero or more characters at the **beginning of str** match this regular expression, return a corresponding match object.
  Return None if the string does not match the pattern; note that this is different from a zero-length match.

```
pattern = re.compile(r"(\d{4})")
m=pattern.match("1945 - 2022")
m.groups() # The entire match
```

```
('1945',)
```

Regex using Python

- re.findall(*pattern*, *string*, *flags=0*)
  Return all non-overlapping matches of *pattern* in *string*, as a **list of strings or tuples**. The *string* is scanned left-to-right, and matches are returned in the order found. Empty matches are included in the result.

```
text="17 Agustus 1945 - 2022"
re.findall(r"(\d{4})", text)
```

```
['1945', '2022']
```

```
#extract adverb
text = "He was carefully disguised but captured quickly by police."
re.findall(r"\w+ly\b", text)
```

```
['carefully', 'quickly']
```

Regex using Python

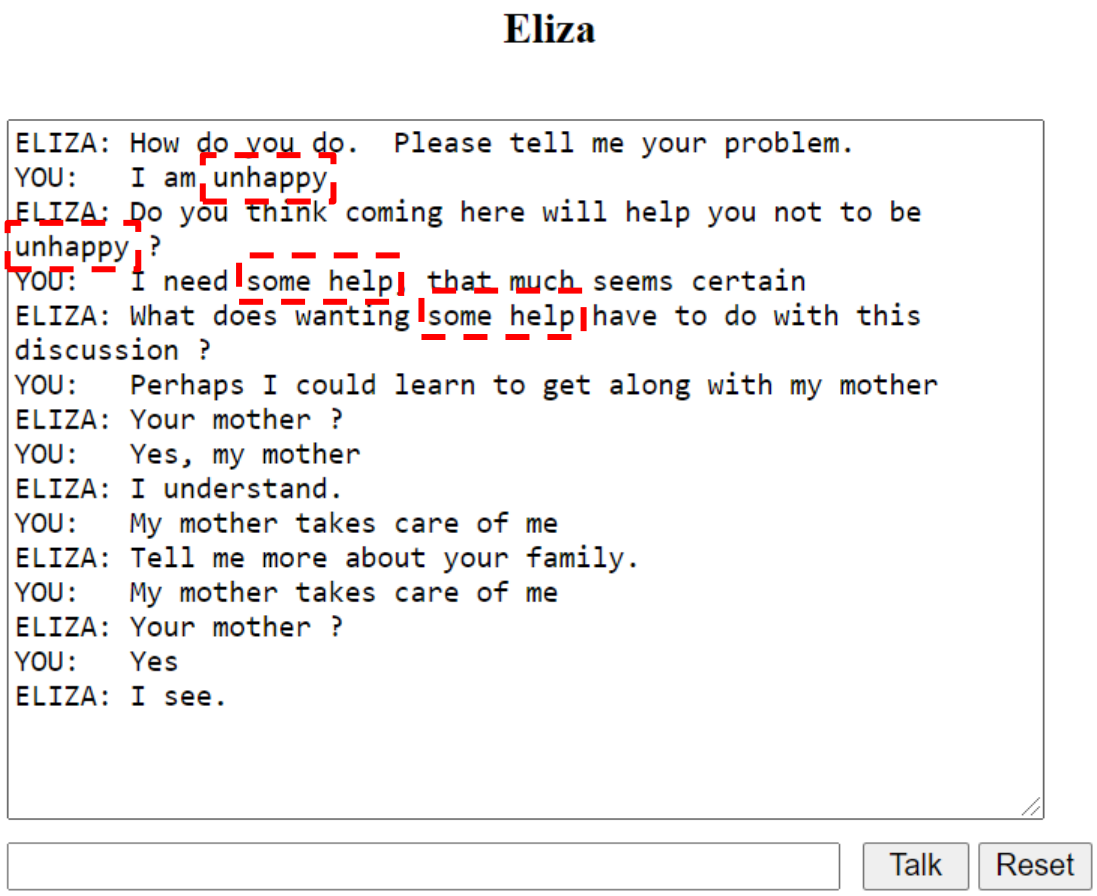
```
In [32]: pattern = re.compile(r"(\d{4})")
         pattern.search("17 Agustus 1945 - 2022") #Scan through string looking for the first location whe

Out[32]: <re.Match object; span=(11, 15), match='1945'>

In [16]: pattern.match("17 Agustus 1945") #If zero or more characters at the beginning of string match th

In [34]: m=pattern.match("1945 - 2022")
         m.groups() # The entire match

Out[34]: ('1945',)

In [36]: text="17 Agustus 1945 - 2022"
         re.findall(r"(\d{4})", text)

Out[36]: ['1945', '2022']

In [35]: #extract adverb
         text = "He was carefully disguised but captured quickly by police."
         re.findall(r"\w+ly\b", text)

Out[35]: ['carefully', 'quickly']
```

https://docs.python.org/3/library/re.html

Baca: https://www.guru99.com/python-regular-expressions-complete-tutorial.html

Eliza, simple pattern-based chatbot

ELIZA uses **pattern matching** to recognize phrases like “I need X” and translate them into suitable outputs like
“What does wanting X have to do with this discussion ?”
“What would it mean to you if you got X?”.



https://www.masswerk.at/elizabot/
http://psych.fullerton.edu/mbirnbaum/psych101/eliza.htm

Weizenbaum, J. (1966). ELIZA—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, *9*(1), 36-45.

Eliza-like Chatbot using Regex

```
import re

def findAnswer(teks):
    t=teks.upper()
    knowledge_base={".* I'M (DEPRESSED|SAD) .*":"I AM SORRY TO HEAR YOU ARE X",
          ".* MY (.*)":"YOUR X",
          ".* ALL .*":"IN WHAT WAY",
          ".* ALWAYS .*":"CAN YOU THINK OF A SPECIFIC EXAMPLE"
         }
    notFound="Can you repeat your question ?"
    for key in knowledge_base:
        m=re.match(key, t)
        if m:
           answer=knowledge_base[key]
           len_groups = len(m.groups())
           if (len_groups==0):
               return answer
           else:
               X=m.group(1)
               answer=answer.replace("X",X)
               if " ME " in answer:
                  answer=answer.replace(" ME "," YOU ")
               return answer
    return notFound
```

```
listQ=["Men are all alike",
       "They're always bagging us about something or other",
       "Well, my boyfriend made me come here",
       "He says I'm depressed much of the time."
      ]

for q in listQ:
    print("User: ",q)
    print("Bot: ",findAnswer(q))
```

```
User:  Men are all alike
Bot:  IN WHAT WAY
User:  They're always bagging us about something or other
Bot:  CAN YOU THINK OF A SPECIFIC EXAMPLE
User:  Well, my boyfriend made me come here
Bot:  YOUR BOYFRIEND MADE YOU COME HERE
User:  He says I'm depressed much of the time.
Bot:  I AM SORRY TO HEAR YOU ARE DEPRESSED
```