From Regular Expressions to Regular Languages

Phúc Phạm

June 10, 2025

Outline

Regex basics

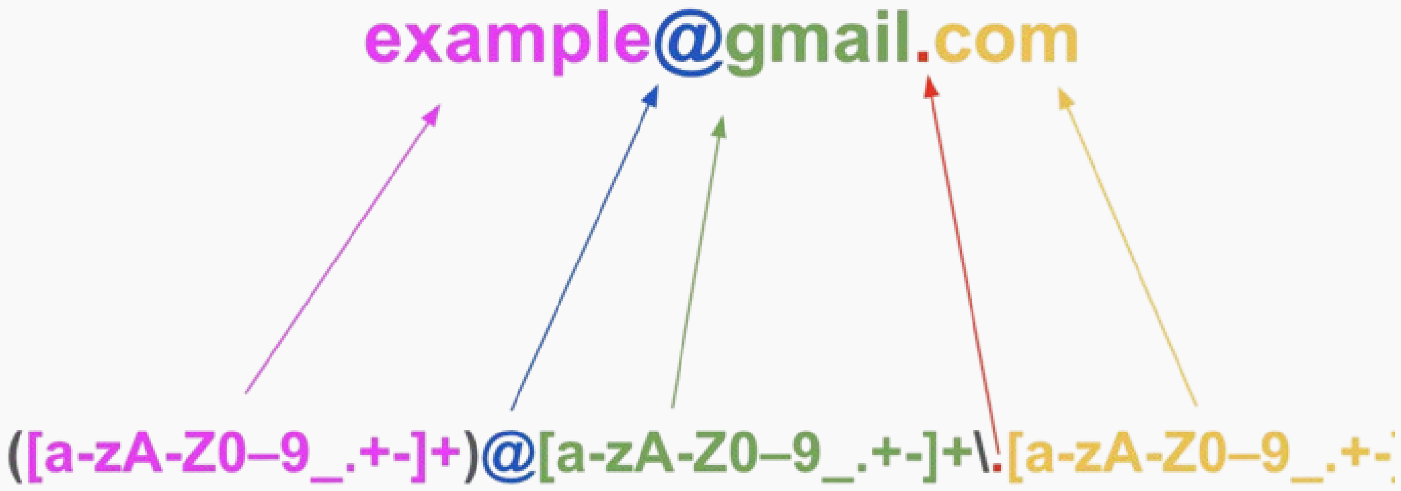
What is a regular expression?

What [a-z]{2} [^A-Z]+ (\w*)\?

What is regular expression?

WHENEVER I LEARN A NEW SKILL I CONCOCT ELABORATE FANTASY SCENARIOS WHERE IT LETS ME SAVE THE DAY.
OH NO! THE KILLER MUST HAVE FOLLOWED HER ON VACATION!
BUT TO FIND THEM WE'D HAVE TO SEARCH THROUGH 200 MB OF EMAILS LOOKING FOR SOMETHING FORMATTED LIKE AN ADDRESS!
IT'S HOPELESS!
EVERYBODY STAND BACK.
I KNOW REGULAR EXPRESSIONS.
PERL!
TAP TAP

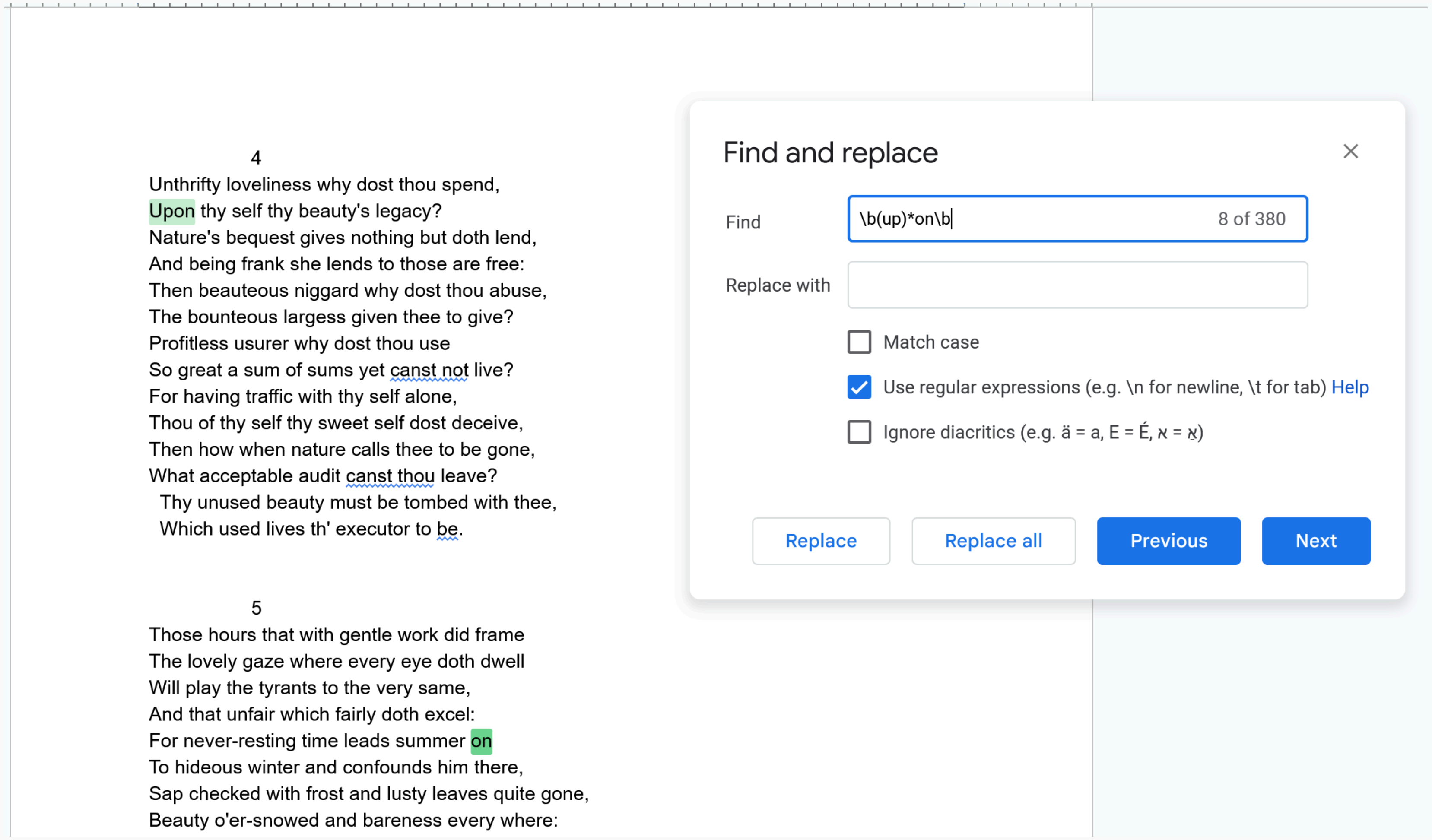
Regex usecases



1. Text search

2. Text validation

Where can I use regex?



4
Unthrifty loveliness why dost thou spend,
Upon thy self thy beauty's legacy?
Nature's bequest gives nothing but doth lend,
And being frank she lends to those are free:
Then beauteous niggard why dost thou abuse,
The bounteous largess given thee to give?
Profitless usurer why dost thou use
So great a sum of sums yet canst not live?
For having traffic with thy self alone,
Thou of thy self thy sweet self dost deceive,
Then how when nature calls thee to be gone,
What acceptable audit canst thou leave?
  Thy unused beauty must be tombed with thee,
  Which used lives th' executor to be.

5
Those hours that with gentle work did frame
The lovely gaze where every eye doth dwell
Will play the tyrants to the very same,
And that unfair which fairly doth excel:
For never-resting time leads summer on
To hideous winter and confounds him there,
Sap checked with frost and lusty leaves quite gone,
Beauty o'er-snowed and bareness every where:

Find and replace

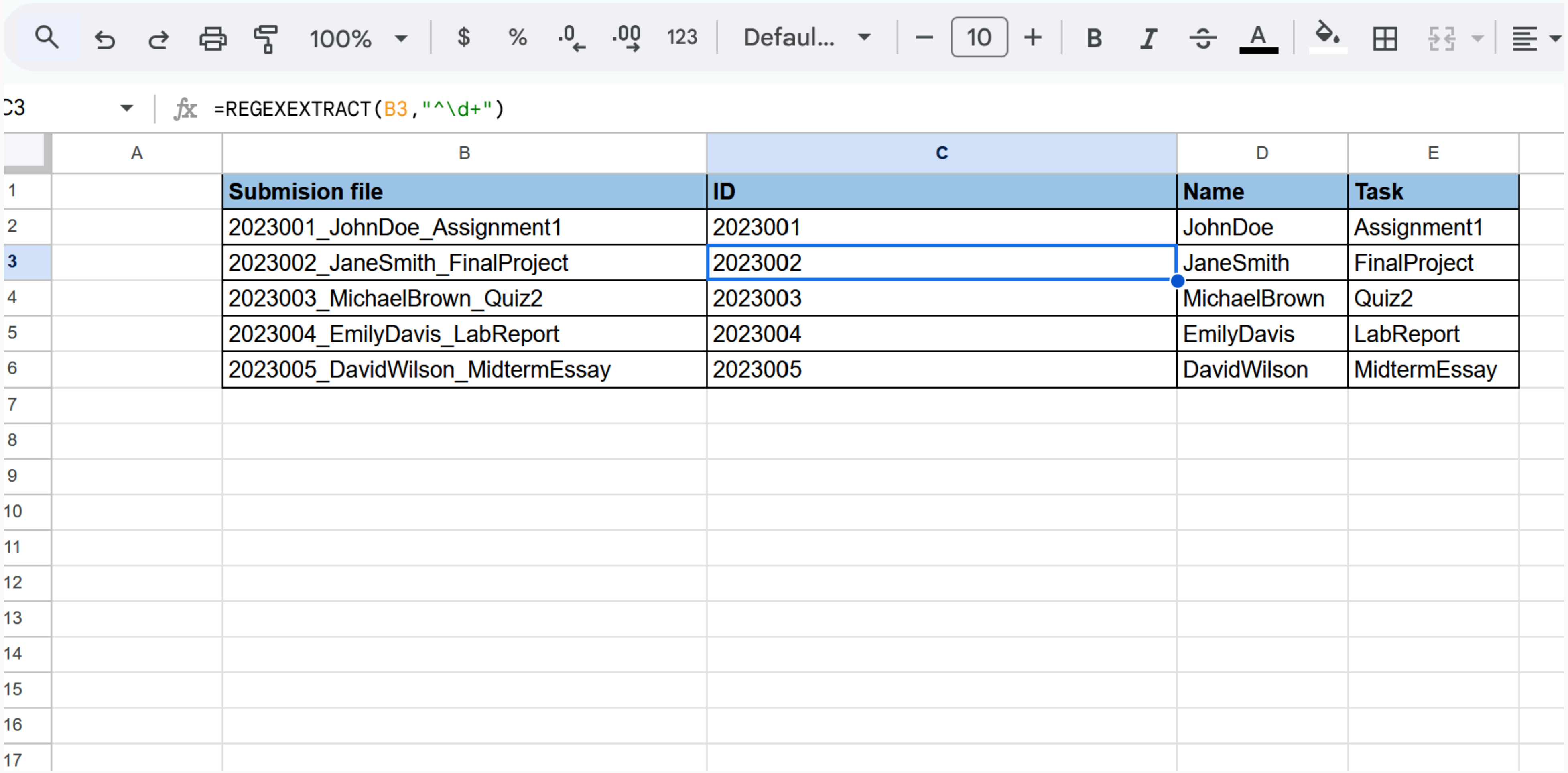
Find: `\b(up)*on\b` 8 of 380

Replace with:

☐ Match case
☑ Use regular expressions (e.g. \n for newline, \t for tab) Help
☐ Ignore diacritics (e.g. ä = a, E = É, א = אָ)

Replace | Replace all | Previous | Next

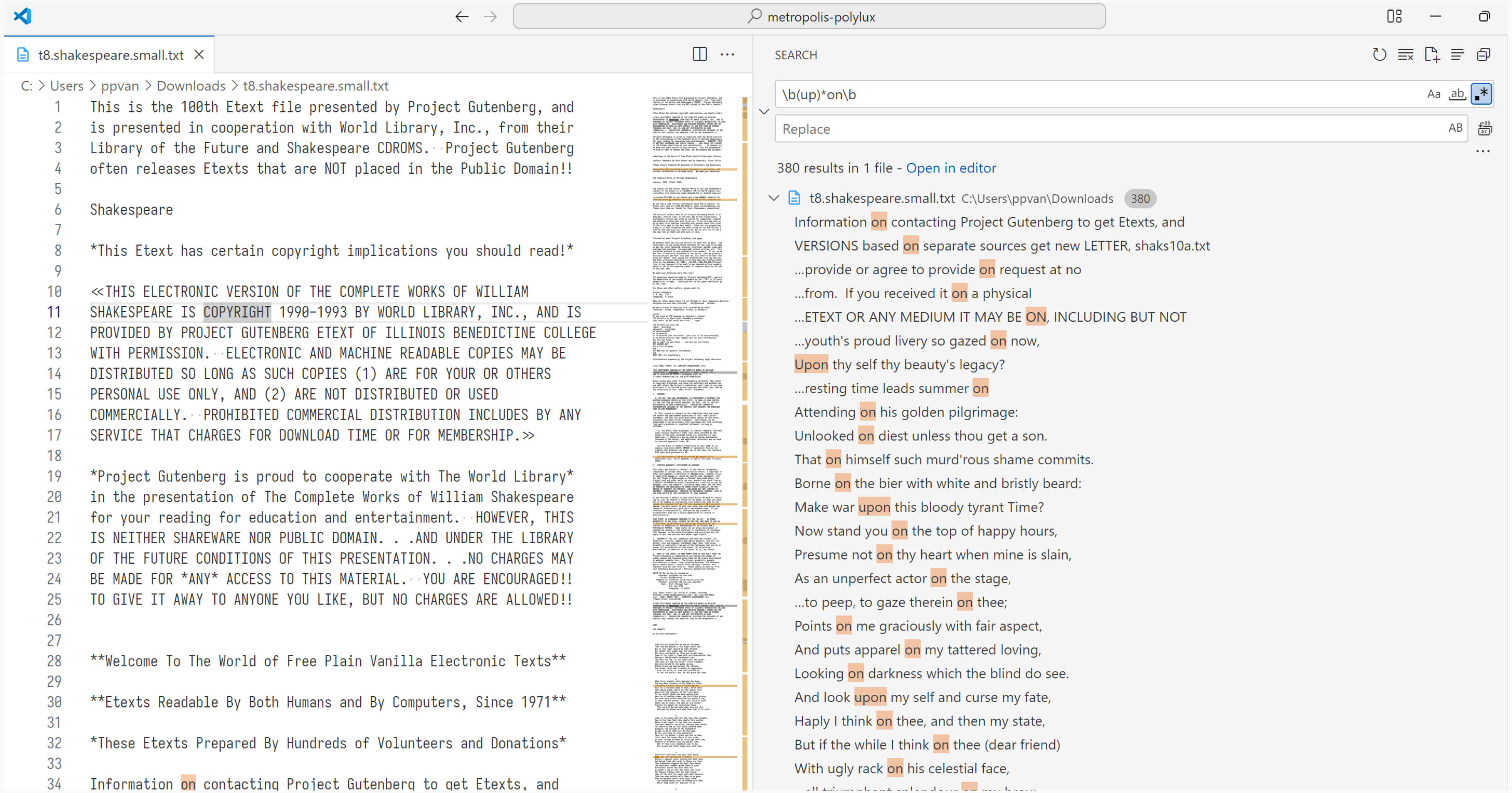
Where can I use regex?



C3 `=REGEXEXTRACT(B3,"^\d+")`

	A	B	C	D	E
1		Submision file	ID	Name	Task
2		2023001_JohnDoe_Assignment1	2023001	JohnDoe	Assignment1
3		2023002_JaneSmith_FinalProject	2023002	JaneSmith	FinalProject
4		2023003_MichaelBrown_Quiz2	2023003	MichaelBrown	Quiz2
5		2023004_EmilyDavis_LabReport	2023004	EmilyDavis	LabReport
6		2023005_DavidWilson_MidtermEssay	2023005	DavidWilson	MidtermEssay
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					

Where can I use regex?



metropolis-polylux

t8.shakespeare.small.txt

C: > Users > ppvan > Downloads > t8.shakespeare.small.txt

```
This is the 100th Etext file presented by Project Gutenberg, and
is presented in cooperation with World Library, Inc., from their
Library of the Future and Shakespeare CDROMS.  Project Gutenberg
often releases Etexts that are NOT placed in the Public Domain!!

Shakespeare

*This Etext has certain copyright implications you should read!*

<<THIS ELECTRONIC VERSION OF THE COMPLETE WORKS OF WILLIAM
SHAKESPEARE IS COPYRIGHT 1990-1993 BY WORLD LIBRARY, INC., AND IS
PROVIDED BY PROJECT GUTENBERG ETEXT OF ILLINOIS BENEDICTINE COLLEGE
WITH PERMISSION.  ELECTRONIC AND MACHINE READABLE COPIES MAY BE
DISTRIBUTED SO LONG AS SUCH COPIES (1) ARE FOR YOUR OR OTHERS
PERSONAL USE ONLY, AND (2) ARE NOT DISTRIBUTED OR USED
COMMERCIALLY.  PROHIBITED COMMERCIAL DISTRIBUTION INCLUDES BY ANY
SERVICE THAT CHARGES FOR DOWNLOAD TIME OR FOR MEMBERSHIP.>>

*Project Gutenberg is proud to cooperate with The World Library*
in the presentation of The Complete Works of William Shakespeare
for your reading for education and entertainment.  HOWEVER, THIS
IS NEITHER SHAREWARE NOR PUBLIC DOMAIN. . .AND UNDER THE LIBRARY
OF THE FUTURE CONDITIONS OF THIS PRESENTATION. . .NO CHARGES MAY
BE MADE FOR *ANY* ACCESS TO THIS MATERIAL.  YOU ARE ENCOURAGED!!
TO GIVE IT AWAY TO ANYONE YOU LIKE, BUT NO CHARGES ARE ALLOWED!!


**Welcome To The World of Free Plain Vanilla Electronic Texts**

**Etexts Readable By Both Humans and By Computers, Since 1971**

*These Etexts Prepared By Hundreds of Volunteers and Donations*

Information on contacting Project Gutenberg to get Etexts, and
```

SEARCH

```
\b(up)*on\b
```

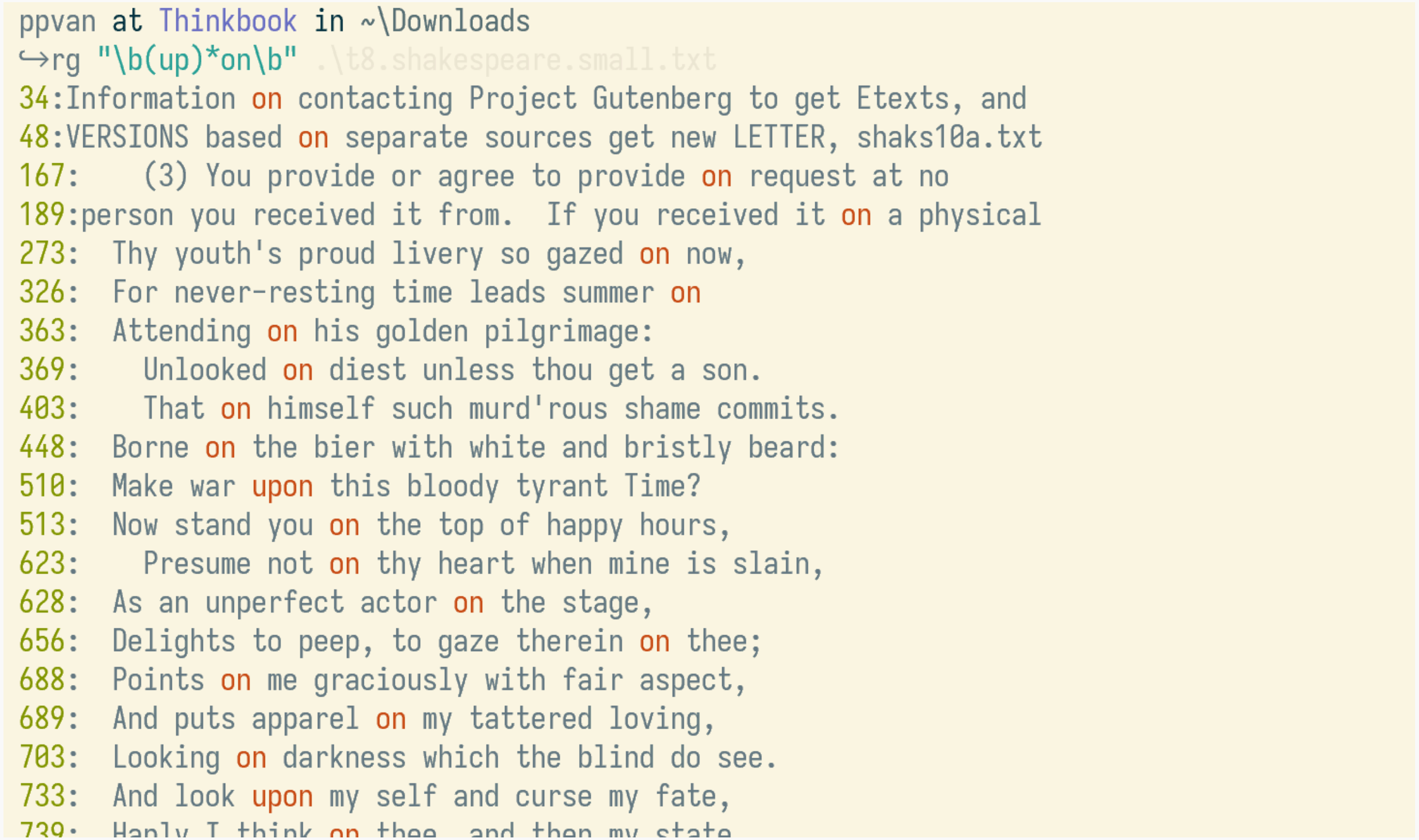
Replace

380 results in 1 file - Open in editor

t8.shakespeare.small.txt C:\Users\ppvan\Downloads 380

```
Information on contacting Project Gutenberg to get Etexts, and
VERSIONS based on separate sources get new LETTER, shaks10a.txt
...provide or agree to provide on request at no
...from.  If you received it on a physical
...ETEXT OR ANY MEDIUM IT MAY BE ON, INCLUDING BUT NOT
...youth's proud livery so gazed on now,
Upon thy self thy beauty's legacy?
...resting time leads summer on
Attending on his golden pilgrimage:
Unlooked on diest unless thou get a son.
That on himself such murd'rous shame commits.
Borne on the bier with white and bristly beard:
Make war upon this bloody tyrant Time?
Now stand you on the top of happy hours,
Presume not on thy heart when mine is slain,
As an unperfect actor on the stage,
...to peep, to gaze therein on thee;
Points on me graciously with fair aspect,
And puts apparel on my tattered loving,
Looking on darkness which the blind do see.
And look upon my self and curse my fate,
Haply I think on thee, and then my state,
But if the while I think on thee (dear friend)
With ugly rack on his celestial face,
```

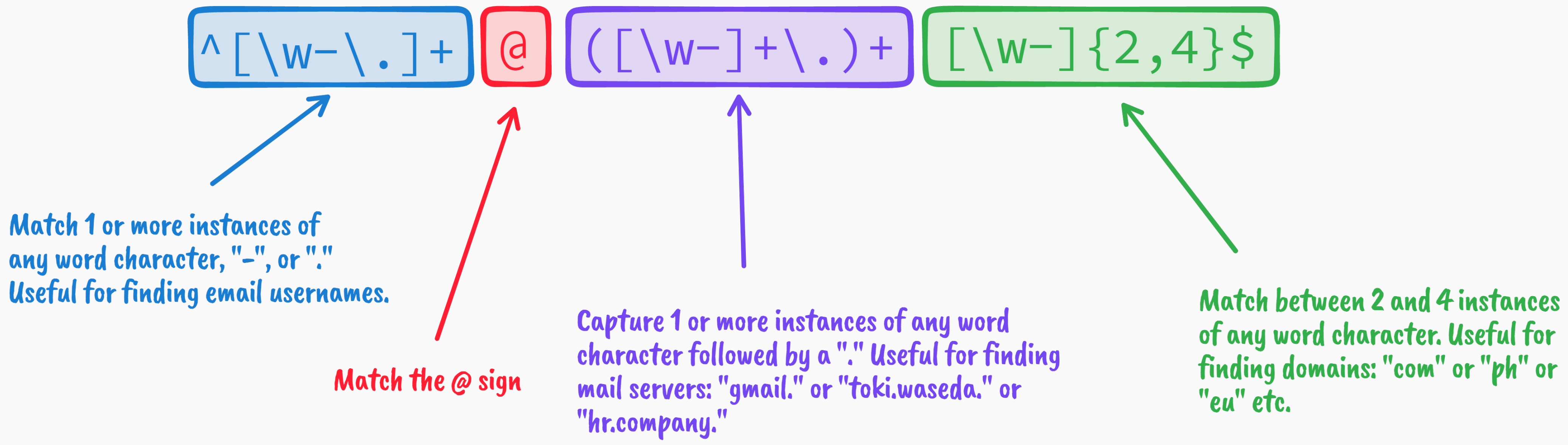
Where can I use regex?



```
ppvan at Thinkbook in ~\Downloads
↪rg "\b(up)*on\b" .\t8.shakespeare.small.txt
34:Information on contacting Project Gutenberg to get Etexts, and
48:VERSIONS based on separate sources get new LETTER, shaks10a.txt
167:    (3) You provide or agree to provide on request at no
189:person you received it from.  If you received it on a physical
273:  Thy youth's proud livery so gazed on now,
326:  For never-resting time leads summer on
363:  Attending on his golden pilgrimage:
369:    Unlooked on diest unless thou get a son.
403:    That on himself such murd'rous shame commits.
448:  Borne on the bier with white and bristly beard:
510:  Make war upon this bloody tyrant Time?
513:  Now stand you on the top of happy hours,
623:    Presume not on thy heart when mine is slain,
628:  As an unperfect actor on the stage,
656:  Delights to peep, to gaze therein on thee;
688:  Points on me graciously with fair aspect,
689:  And puts apparel on my tattered loving,
703:  Looking on darkness which the blind do see.
733:  And look upon my self and curse my fate,
739:  Haply I think on thee, and then my state
```

Regex syntax by example

[1]



```
^[\w-\.]+@([\w-]+\.)+[\w-]{2,4}$
```

Match 1 or more instances of any word character, "-", or "." Useful for finding email usernames.

Match the @ sign

Capture 1 or more instances of any word character followed by a "." Useful for finding mail servers: "gmail." or "toki.waseda." or "hr.company."

Match between 2 and 4 instances of any word character. Useful for finding domains: "com" or "ph" or "eu" etc.

[1] https://regexone.com/

Regex tips in practice

Use anchor (^$) when possible

Regex: /treature/

treature here, treature there, everywhere treature

Regex: /^treature/

treature here, treature there, everywhere treature

Regex: /treature$/

treature here, treature there, everywhere treature

/.*/ is rarely what you want

Regex: `/user_id="(.*)"/`

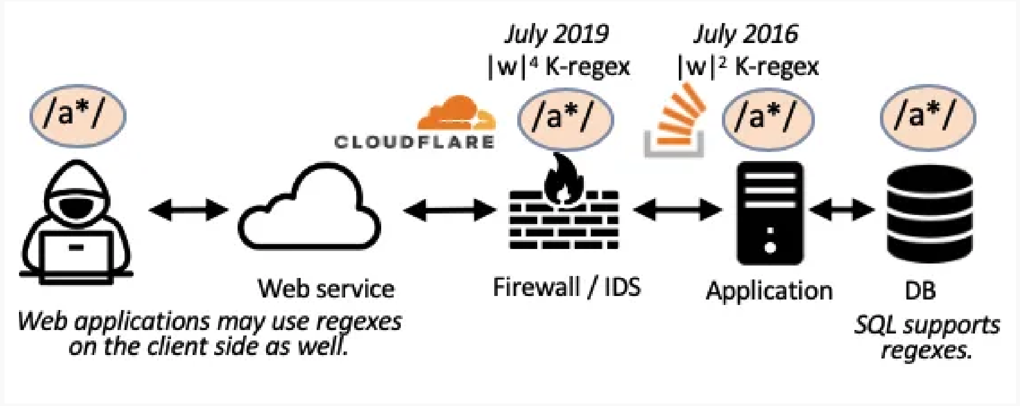
```
2024-01-15 user_id="123" action=login error="Invalid password"
ip=192.168.1.100 session="abc123"
```

Regex: `/user_id="([^"])"/`

```
2024-01-15 user_id="123" action=login error="Invalid password"
ip=192.168.1.100 session="abc123"
```

Regex is code

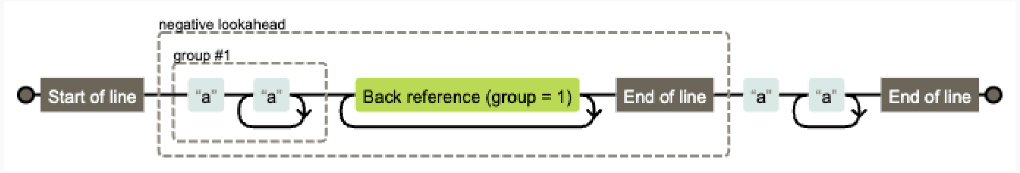
- Catastrophic backtracking
- Regular Expression Denial of Service



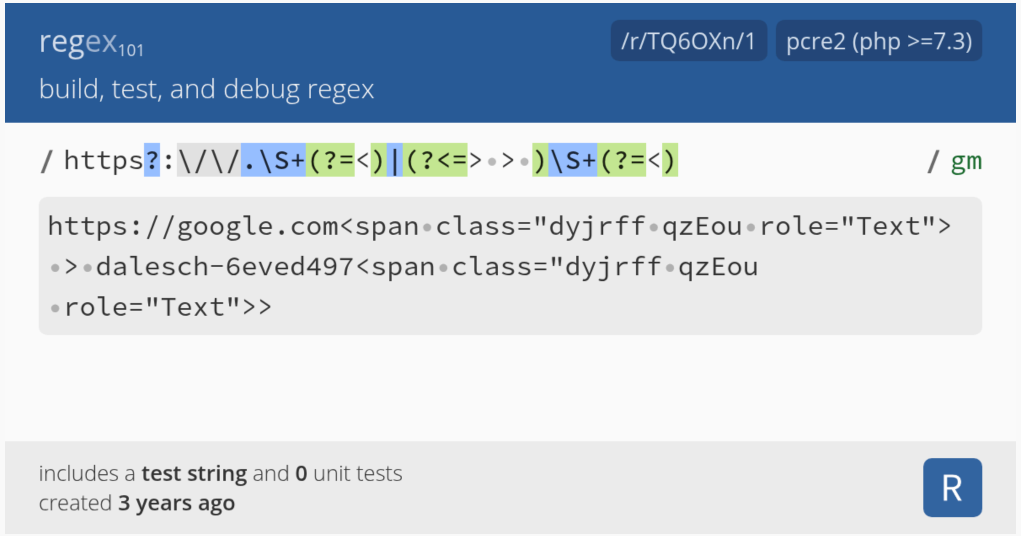
2

[2]https://blog.cloudflare.com/details-of-the-cloudflare-outage-on-july-2-2019/

Use visualization tools



regexper.com



```
/ https?:\/\/.\S+(?=<)|(?<=> > )\S+(?=<) / gm
```

```
https://google.com<span class="dyjrff qzEou role="Text">
 > dalesch-6eved497<span class="dyjrff qzEou
 role="Text">>
```

includes a **test string** and **0** unit tests
created **3 years ago**

regex101.com

Regular Languages

What is regular languages?

$$\boxed{\texttt{/a(bb)+a/}} \longrightarrow$$

```
abba
abbbba
abbbbbba
abbbbbbbba
abbbbbbbbbba
.....
```

$$L = \{a(\texttt{bb})^n a \mid n \geq 1\}$$

Finite Automata?

$\boxed{\texttt{/a(bb)+a/}}$ $\longleftarrow$

```
abba
abbbba
abbbbbba
abbbbbbbba
abbbbbbbbbba
.....
```

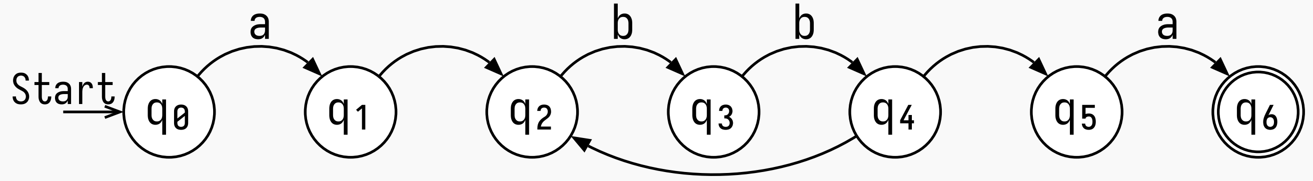
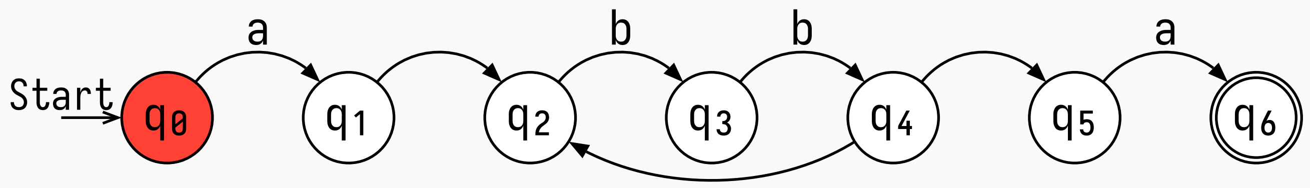
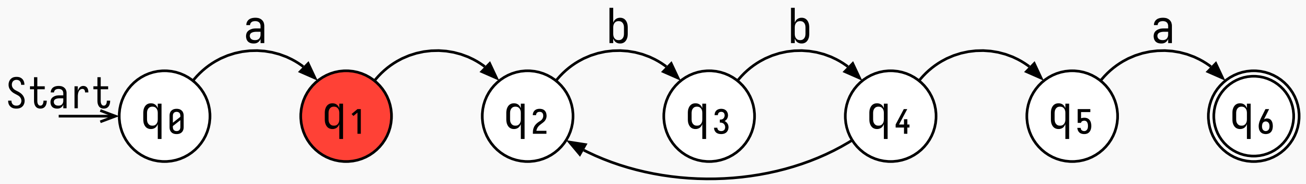


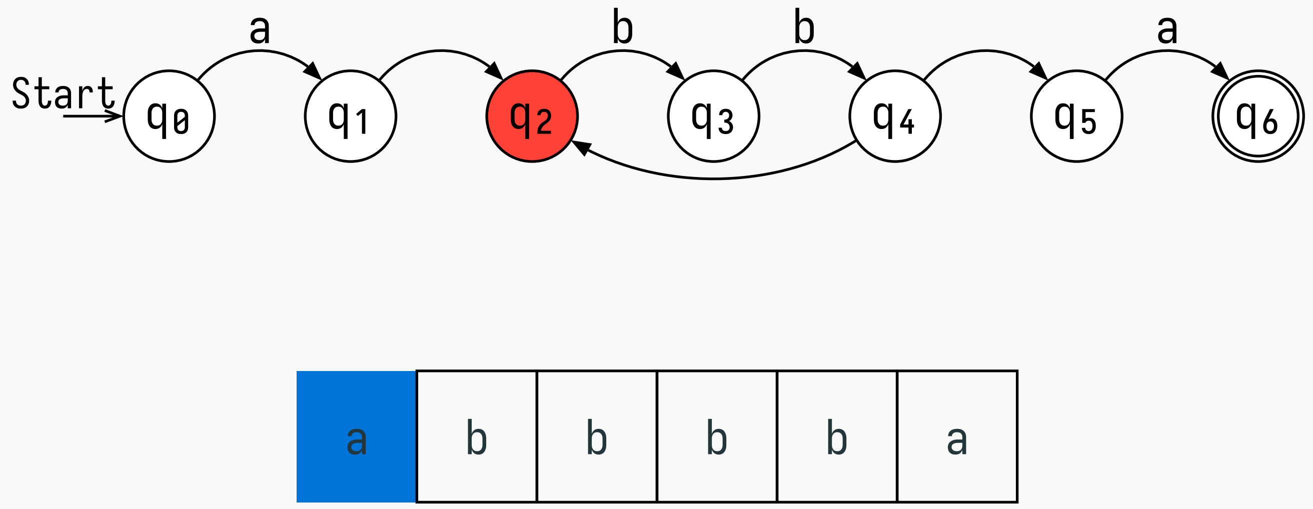
Figure 5: /a(bb)+a/ finite automata



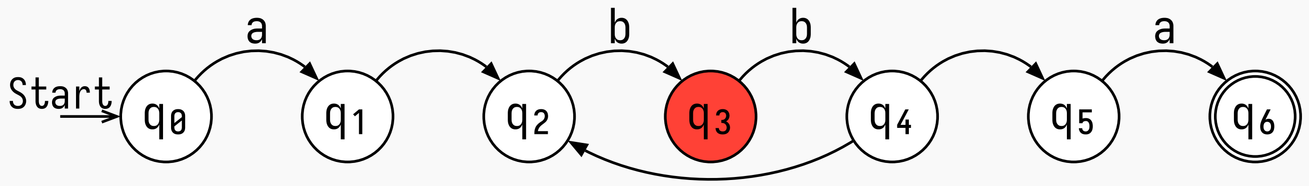
a	b	b	b	b	a

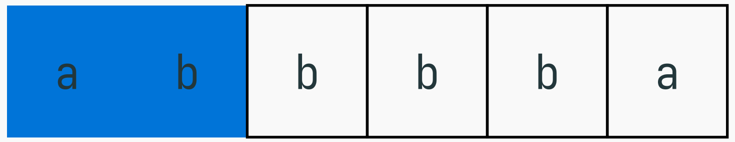


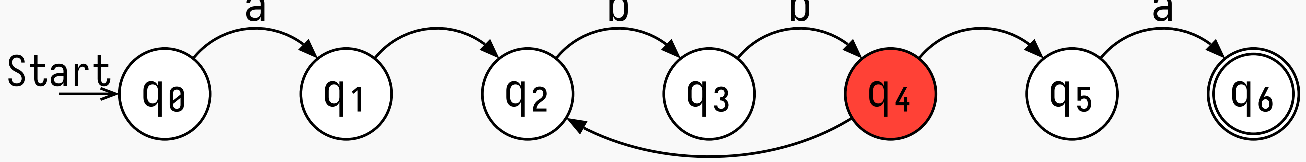
a	b	b	b	b	a



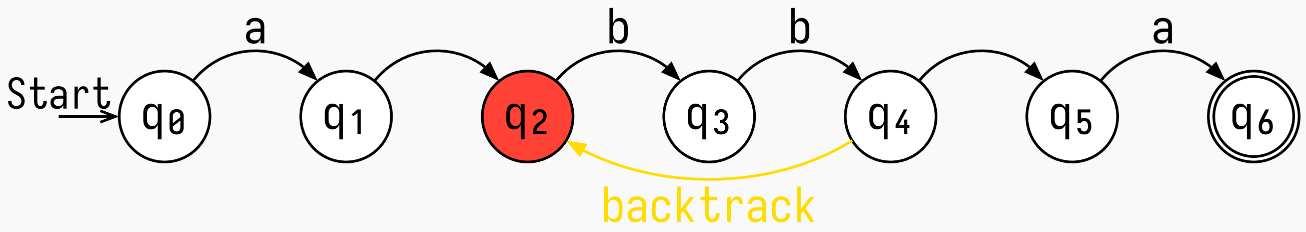
Start
q0
a
q1
q2
b
q3
b
q4
q5
a
q6
a
b
b
b
b
a

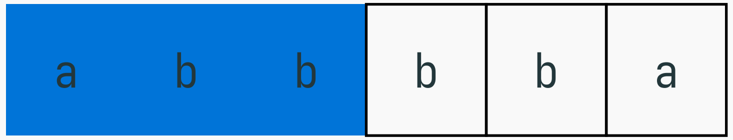

Start
q0
a
q1
q2
b
q3
b
q4
q5
a
q6

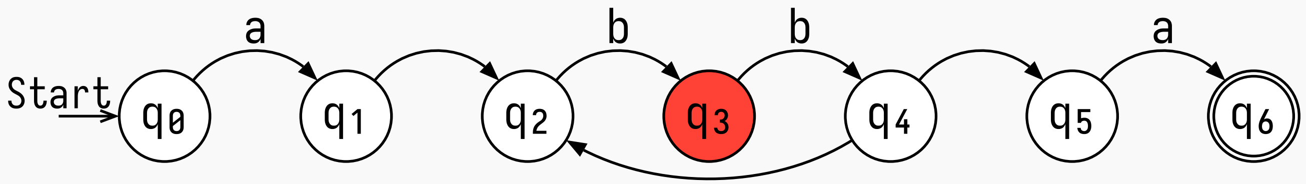

a
b
b
b
b
a

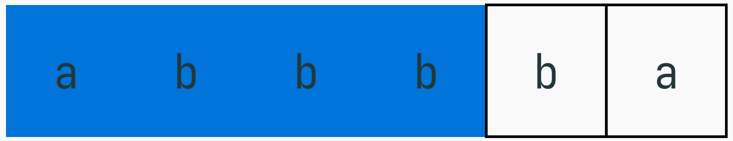


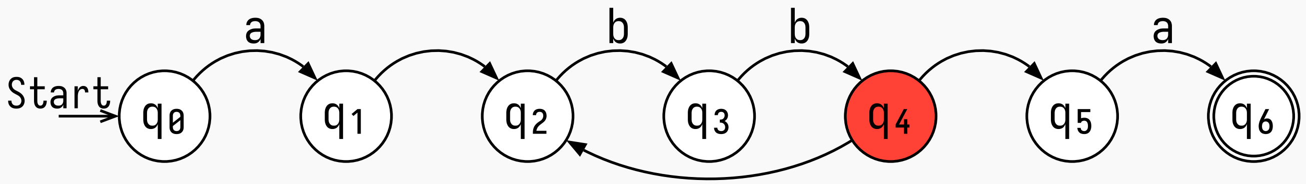
a	b	b	b	b	a

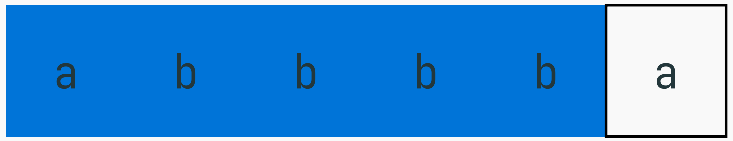

Start
q0
q1
q2
q3
q4
q5
q6
a
b
b
a
backtrack

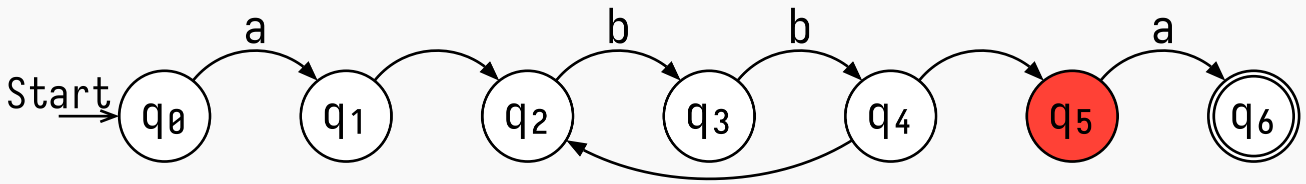

a
b
b
b
b
a

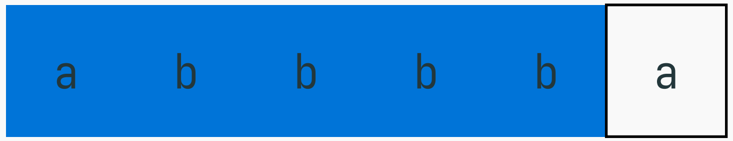

a
b
b
a
Start
q0
q1
q2
q3
q4
q5
q6

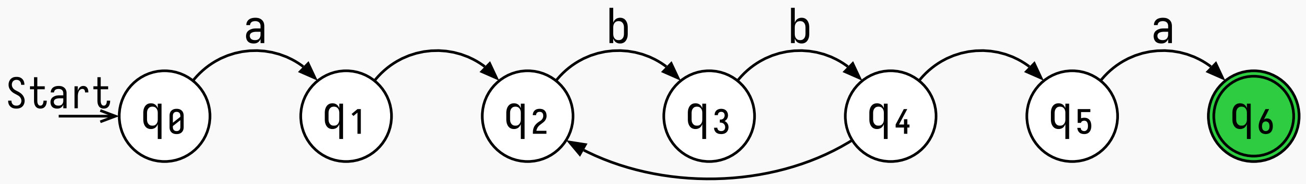

a
b
b
b
b
a


Start
q0
q1
q2
q3
q4
q5
q6
a
b
b
a


a b b b b a


Start
a
b
b
a
q0
q1
q2
q3
q4
q5
q6


a
b
b
b
b
a



a b b b b a

Catastrophic backtracking

Regex repetitions (e.g ?*+) causes backtracking in regex

→ O(2^n) complexity

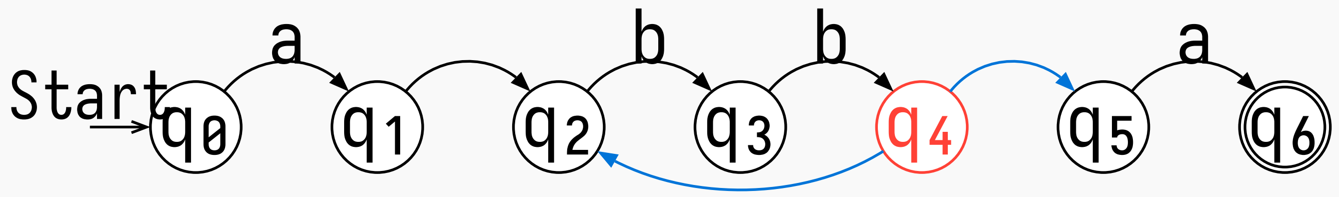


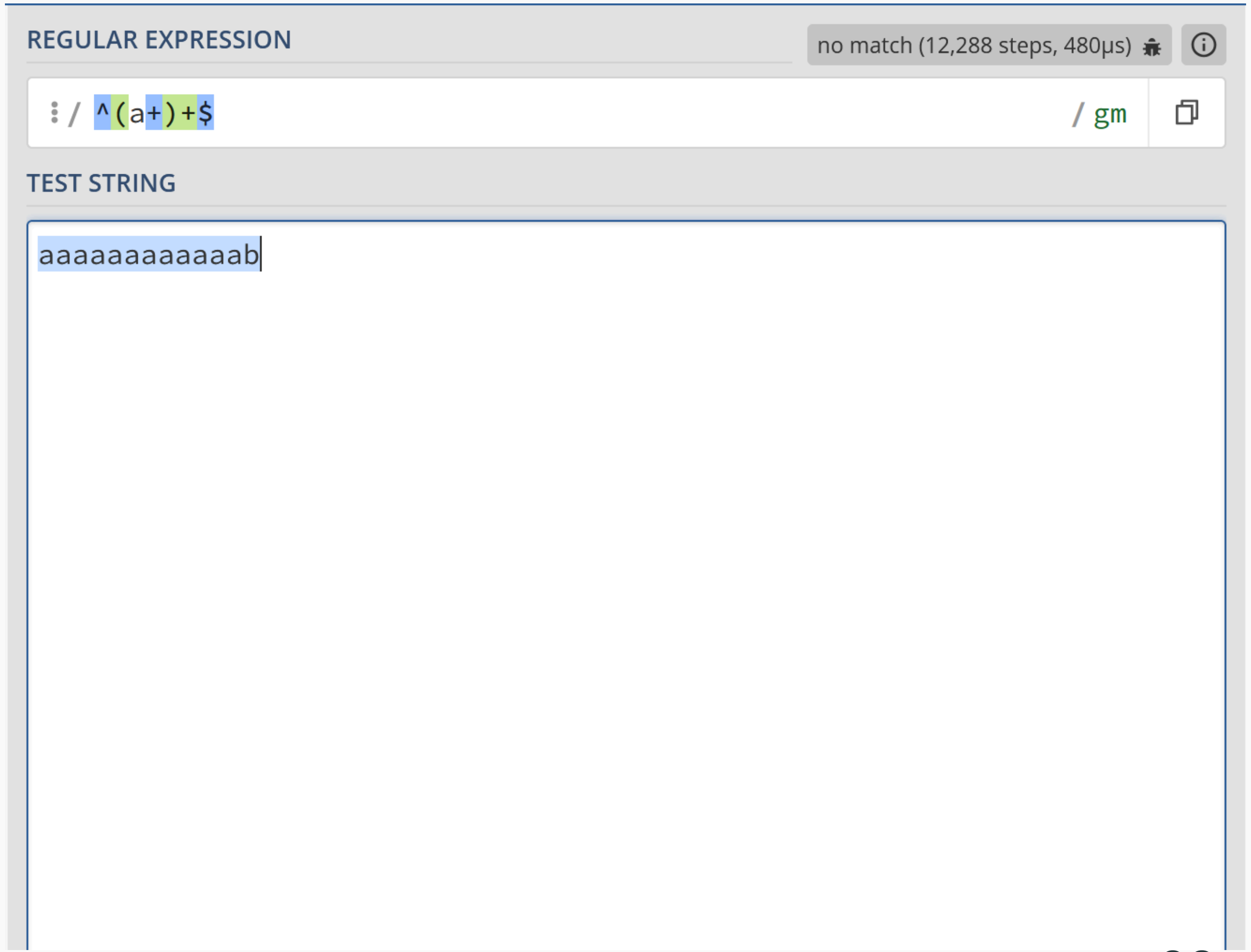
Figure 6: /a(bb)+a/ state machine

Catastrophic Backtracking

Pattern: ^(a+)+$

Text: aaaaaaaaaaaab

→ ~12.000 steps



REGULAR EXPRESSION

no match (12,288 steps, 480μs)

```
/ ^(a+)+$ / gm
```

TEST STRING

```
aaaaaaaaaaaab
```

Introduction to parser

Why do we need parser?

Regex can’t “count”

- html: <tag/>
- json: {}

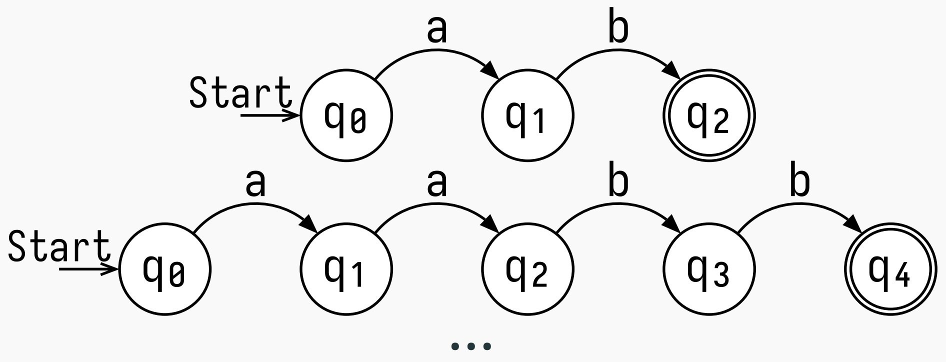


Figure 7: NFA can't represent {ab, aabb, aaabbb, ...}

Regex vs Parsers

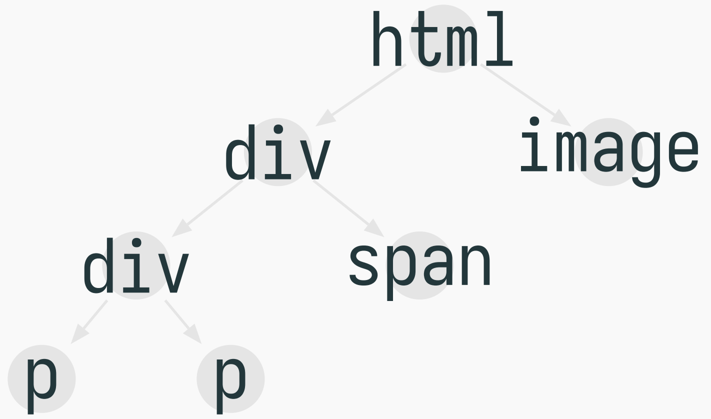
Parsers are more powerful and intuitive

Regex

```
/<p\s+(?:[^>]*\s+)?
class\s*=\s*["\'](["^"\']*)["\']
[^>]*>((?:[^<]|<(?!/p>))*)</p>/
```

Match all p tag with text

HTML tree



Parsers required more understanding

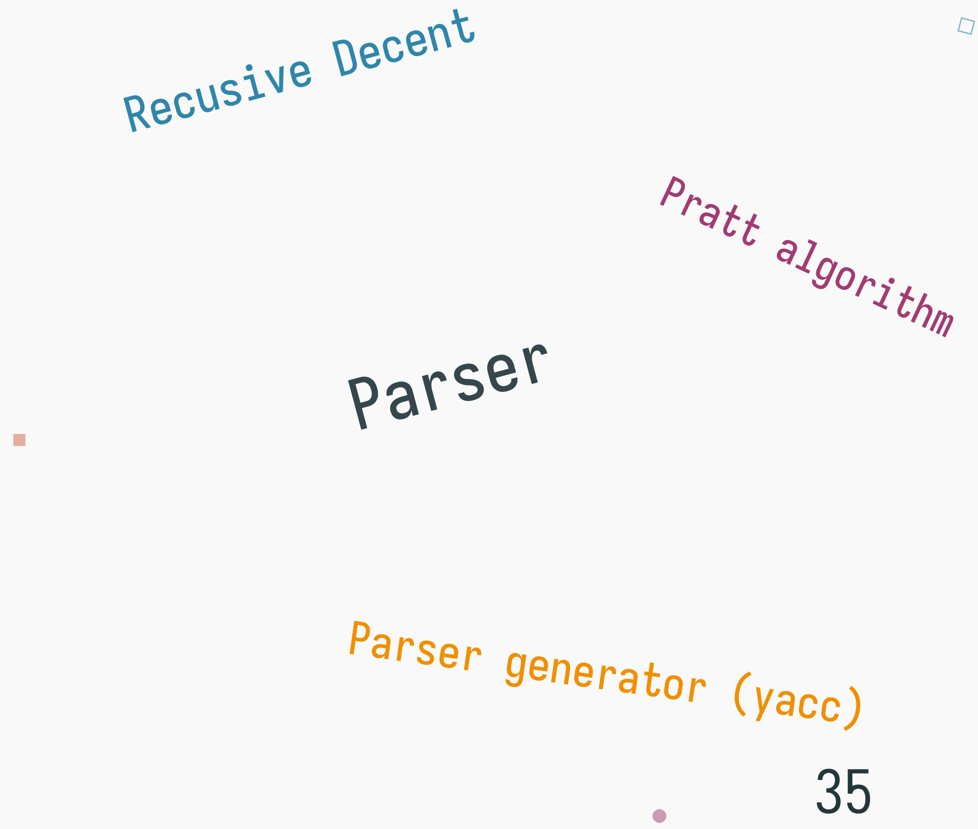
```
from lark import Lark
grammar = """
    start: expr
    expr: term ("+" term | "-" term)*
    term: factor ("*" factor | "/" factor)*
    factor: NUMBER | "(" expr ")"
    NUMBER: /[0-9]+/
"""


parser = Lark(grammar)
tree = parser.parse("2 + 3 * 4")
```

Table 1: Parser that match simple math

Parser usecases

- Compilers & Interpreters
- Data validation: JSON, XML
- Configuration files: YAML, TOML



Summary

- Regex is a powerful tools to match “flat” structure extremely fast
- Modern regex is complicated and require careful consideration
- When regex become too hard, consider a parser

Questions?