

REGEX

Written by Tyler Weiss 18 MAR 2024

Exercise 4

Task #1

1. Use boo as a literal in your regex
2. Discover what it selects

REGULAR EXPRESSION

3 matches (12 steps, 3.5ms)

/ boo

/ gm

TEST STRING

boo^d
boomerang^d
taboo^d
1000^d
1000000^d
0101000010

MATCH INFORMATION

Match 1

0-3

boo

Match 2

4-7

boo

Match 3

16-19

boo

Using the literal string 'boo' as the regex argument matches any literal occurrence of the string as shown above.

Task #2

1. Use 1000 as a literal in your regex
2. Discover what it selects

REGULAR EXPRESSION 3 matches (18 steps, 0.0ms)

/ 1000 / gm

TEST STRING

```
bood
boomerangd
tabood
1000d
10000000d
0101000010
```

MATCH INFORMATION

Match 1	20-24	1000
Match 2	25-29	1000
Match 3	36-40	1000

Using the literal string '1000' as the regex argument matches any literal occurrence of the string as shown above.

Task #3

1. Use `^boo` as a literal in your regex
2. Discover what it selects

REGULAR EXPRESSION 2 matches (14 steps, 0.1ms)

/ ^boo / gm

TEST STRING

```
bood
boomerangd
tabood
1000d
10000000d
0101000010
```

MATCH INFORMATION

Match 1	0-3	boo
Match 2	4-7	boo

Using the anchor '^' specifies that the the regex is looking for a match at the beginning of a string.

The regex argument '^boo' states that the string literal 'boo' should only match if it occurs at the beginning of a string which resulted in the only two matches as shown above.

Task #4

1. Use ^1000 as a literal in your regex
2. Discover what it selects

REGULAR EXPRESSION

2 matches (22 steps, 0.0ms)

:/^1000/gm

TEST STRING

boo
boomerang
taboo
1000
1000000
0101000010

MATCH INFORMATION

Match 120-241000

Match 225-291000

Using the anchor '^' specifies that the the regex is looking for a match at the beginning of a string. The regex argument '^1000' states that the string literal '1000' should only match if it occurs at the beginning of a string which resulted in the only two matches as shown above.

Task #5

1. Use ^boo\$ as a literal in your regex
2. Discover what it selects

REGULAR EXPRESSION1 match (14 steps, 0.0ms)

/ ^boo\$ / gm

TEST STRING

boo
boomerang
taboo
1000
1000000
0101000010

MATCH INFORMATION

Match 10-3boo

Using the anchor '^' in conjunction with the anchor '\$' specifies that the regex contained from the beginning to the end of a string. The regex argument '^boo\$' states that the string literal 'boo' should only match if it is at the beginning and end of the string at the same time. Thus the only match would be a 'boo' as shown above.

Task #6

1. Use ^1000\$ as a literal in your regex
2. Discover what it selects

REGULAR EXPRESSION1 match (22 steps, 0.1ms)

/ ^1000\$ / gm

TEST STRING

boo
boomerang
taboo
1000
1000000
0101000010

MATCH INFORMATION

Match 120-241000

Using the anchor '^' in conjunction with the anchor '\$' specifies that the regex contained from the beginning to the end of a string. The regex argument '^1000\$' states that the string literal '1000' should only match if it is at the beginning and end of the string at the same time. Thus the only match would be a '1000' as shown above.

Exercise 5

Task #1

1. Only use the following
 1. Use ^
 2. Use \d
 3. Use \D
 4. Use \s
 5. Use \w
2. What is the fastest way?
3. Select the whole TEST STRING as a single match.

REGULAR EXPRESSION

1 match (3 steps, 0.0ms)

/ \d+\D+/ gm

TEST STRING

22 •Acadia •Avenue, •London, •East •End

MATCH INFORMATION

Match 1 0-34 22 •Acadia •Avenue, •London, •East •End

Using only the options listed above, the quickest way to match the the entire string would is to use the class 'd' with the quantifier '+' to match one or more digits. Then use the class 'D' with the quantifier '+' to match one or more characters that are non-digits. The above capture shows that this only takes 3 steps to match the entire string. The below capture shows a more specific way to match the string but will also be slower.

REGULAR EXPRESSION

1 match (15 steps, 0.0ms)

/ ^\d+\s\w+\s\w+\D\s\w+\D\s\w+\s\w+/ gm

TEST STRING

22 •Acadia •Avenue, •London, •East •End

Exercise 6

Task #1 and Task #2

1. Use only commas as literal characters.
2. Only using the Metacharacters of \d and \D
3. Select the whole TEST STRING as a single match.

REGULAR EXPRESSION

1 match (3 steps, 0.1ms)

⋮ / \D+\d+

/ gm

📋

TEST STRING

John•Q. •Adams, •Denver, •Colorado, •80123

MATCH INFORMATION

▼

Match 1

0-38

John•Q. •Adams, •Denver, •Colorado, •80123

📋

Using the class '\D' with '+' matches all non-digit characters one or more times to include spaces and commas. The class '\d' with '+' matches one or digits creating a full match for the string in the quickest possible way.

Exercise 7

Task #1

1. Create a single REGEX that selects both words
2. Select the whole TEST STRING as a single match

REGULAR EXPRESSION

1 match (4 steps, 0.1ms)

⋮ / \w+\s+\w+

/ gm

📋

TEST STRING

Flavour • ↵
↵
Flavor

MATCH INFORMATION

▼

Match 1

0-16

Flavour • ↵
↵
Flavor

📋

The above capture uses the class '\w' and '\s' with the quantifier appended to match the first and last words of any length with one or more space characters in between to include carriage returns.

Exercise 8

Task #1

1. Create a character class
2. ONLY use that class to select the string
3. Select the whole TEST STRING as a single match

REGULAR EXPRESSION

1 match (2 steps, 0.0ms)

/

`[\\w\\s'\\.]+`

/ gm

TEST STRING

Today we're learning regular expressions.

MATCH INFORMATION

Match 1

0-41

Today we're learning regular expressions.

The character class used includes all numbers, letters, commas and apostrophes with the quantifier '+' to match one or more characters resulting in the selection of the entire string.

Exercise 9

Task #1

1. Match all 3 strings with one REGEX
2. The only literals you can use is “is” and “day”
3. Select the whole TEST STRING as a single match
4. Hint..... OPTIONALS are a thing

REGULAR EXPRESSION

1 match (12 steps, 0.1ms)

`/^(is)?[\w\s]+day$/gm`

TEST STRING

Apple is the word of the day
Banana is the word of the day
is the word of the day

MATCH INFORMATION

Match 10-81

Apple is the word of the day
Banana is the word of the day
is the word of the day

The above regex will select all three strings with 'is' being optional at the beginning of the string and 'day' at the end. This expression will also select all three strings as a single match.

Exercise 10

Task #1

1. Match all 3 strings with one REGEX
2. Hint..... OPTIONALS are a thing

REGULAR EXPRESSION

3 matches (59 steps, 0.0ms)

`/^User\s[\w\\]+\slogged\sin$/gm`

TEST STRING

User Tom logged in
User contoso\batman logged in
User contoso\joker logged in

MATCH INFORMATION

Match 10-18

User Tom logged in

Match 219-48

User contoso\batman logged in

Match 349-77

User contoso\joker logged in

Exercise 11

Task #1

1. Match both strings
2. Must use a class

REGULAR EXPRESSION

2 matches (10 steps, 0.0ms)

gr[ae]y

/ gm

TEST STRING

gray^d

grey

MATCH INFORMATION

Match 1

0-4

gray

Match 2

5-9

grey

Using a the class '[ae]' the expression can select both spellings of the strings.

Task #2

1. Match both strings
2. Must use a class

REGULAR EXPRESSION

2 matches (16 steps, 0.0ms)

Advis[eo]r

/ gm

TEST STRING

Adviser^d

Advisor

MATCH INFORMATION

Match 1

0-7

Adviser

Match 2

8-15

Advisor

Using a the class '[eo]' the expression can select both spellings of the strings.

Task #3

1. Match both strings
2. Must use a class
3. And... something else?

REGULAR EXPRESSION

2 matches (18 steps, 0.0ms)

:/ [ae]{1,2}sthetic / gm

TEST STRING

aesthetic^d
esthetic

MATCH INFORMATION			
Match 1	0-9	aesthetic	↑
Match 2	10-18	esthetic	

Using a the class '[ae]' in conjunction with a range of one to two characters the expression can select both spellings of the strings.

Task #4

1. Match both strings
2. Must use a class
3. And... something else?

REGULAR EXPRESSION

2 matches (16 steps, 0.0ms)

:/ analog[ue]* / gm

TEST STRING

analog^d
analogue

MATCH INFORMATION			
Match 1	0-6	analog	↑
Match 2	7-15	analogue	

Using a the class '[ue]' in conjunction with the quantifier '*' to select zero or more characters, the expression can select both spellings of the strings.

Exercise 12

Task #1

1. Match all strings
2. Must use a classes built of base-16 ranges (hexadecimal)

REGULAR EXPRESSION

5 matches (20 steps, 0.0ms)

/ 0x[\dA-Fa-f]+

/ gm

TEST STRING

0x000000^d

0xFFFFF^d

0x000000^d

0x0000DF^d

0x000A00

MATCH INFORMATION

Match 1	0-8	0x000000	
Match 2	9-17	0xFFFFF	
Match 3	18-26	0x000000	
Match 4	27-35	0x0000DF	
Match 5	36-44	0x000A00	

All the above hexadecimal notion uses starts with a '0x' witch can be used as a string literal at the beginning of the expression. Using a the class '[\dA-Fa-f]' and the quantifier '+' the expression can select all characters represented in a hexadecimal format.

Exercise 13

Task #1

1. Match any MAC address
2. Must use a classes built of base-16 ranges (hexadecimal)

The capture below is from the output of the command 'ipconfig /all'

Wireless LAN adapter Wi-Fi:

```
Connection-specific DNS Suffix  . :  
Description . . . . . : Intel(R) Wi-Fi 6 AX201 160MHz  
Physical Address. . . . . : 40-EC-99-C7-67-85  
DHCP Enabled. . . . . : Yes  
Autoconfiguration Enabled . . . . : Yes
```

REGULAR EXPRESSION

1 match (18 steps, 0.1ms)

:/ (?:[\dA-Fa-f]{2}[:-]){5}[\dA-Fa-f]{2}/ gm

TEST STRING

40-EC-99-C7-67-85

MATCH INFORMATION

Match 1 0-17 40-EC-99-C7-67-85

Using the class '[\dA-Fa-f]' will capture any hex character. The class '[:-]' will capture all characters that separate the MAC address octets. So by using a non-capture group that specifies an octet followed by the possible delimiters we can match the first 5 octets with their delimiters. Then the last octet is captured giving a very specific expression for mating MAC addresses.

Task #2

1. Use the the class [A-z]
2. Select most characters with the class

REGULAR EXPRESSION

1 match (87 steps, 0.0ms)

:/ (?:[A-z]+[\s\.,,]{0,2})+

TEST STRING

Tim, Ron and Wallace, knew from their studies that regex could be used
in looking for text within an operating systems and in files in that operating
system.

MATCH INFORMATION

Match 1 0-159 Tim, Ron and Wallace, knew from their studies that regex
could be used
in looking for text within a...

The class '[A-z]' with the quantifier '+' will select the words in the string. Adding the class '[\s\.,,]' for a range of '{0,2}' will also select the space and punctuation characters after each word. With the

quantifier '+' added to the end of the non-capture group the expression will grab one or more instances of the group thus selecting the entire string.

Task #3

1. Only using 2 characters in the expression, select the whole string

The screenshot shows a regex testing interface. At the top, the 'REGULAR EXPRESSION' field contains `/.+`. To the right, a green status bar indicates '1 match (2 steps, 0.0ms)'. Below this, the 'TEST STRING' field contains the text: 'Christian, Tyler, and Reece, didn't like regex at all. But they knew it was important to learn for lots of reasons.' The entire string is highlighted in blue. At the bottom, the 'MATCH INFORMATION' section shows 'Match 1' with a range of '0-115' and the full text of the string.

Only using 2 characters in the expression we have to use the '.' metacharacter which represents all characters. If the quantifier '+' is used the expression will then select all 1 or more of any character. Note, do not use the quantifier '*' because that will select zero or more characters, the string argument and another null string at the end, thus yielding 2 matches.

Task #4

1. Create a regular expression that will match the whole sentence as one match without using `\w` or `\W`.

The screenshot shows a regex testing interface. At the top, the 'REGULAR EXPRESSION' field contains `[\D\d]+\. $`. To the right, a green status bar indicates '1 match (5 steps, 0.1ms)'. Below this, the 'TEST STRING' field contains the text: 'Richard Simmons 1992 album is the greatest hip hop album of all time, including Childish Gambino.' The entire string is highlighted in blue. At the bottom, the 'MATCH INFORMATION' section shows 'Match 1' with a range of '0-97' and the full text of the string.

By using the class `[\D\d]` with the quantifier '+' the expression will select all non-digit characters and

all digits 1 or more times. The '\.\$' anchor was added to the end creating a delimiter, thus the expression would select entire sentences if they end in a period.

Exercise 14

Task #1

1. Create a REGEX that will match
 1. The date
 2. the 4 digit EVID code
 3. PROTIPS
 1. The date can change
 2. The time can change
 3. The EVID can change
 4. Don't use literals for what can change
2. Select the whole TEST STRING as a single match

REGULAR EXPRESSION

1 match (33 steps, 0.0ms)

```
/(?<date>\d{2}\/\d{2}\/\d{4})\s(?<time>\d{2}:\d{2}:\d{2}.\d{4})\sEVID:(?<evid>\d{4})\s-\s(?<msg>.*)
```

TEST STRING

10/11/2008 08:15:00.0154 EVID:4140 -- A 'Extreme failure' has occurred. Your system is been down, your cat has cheezburger!

MATCH INFORMATION			
Match 1	0-122	10/11/2008 08:15:00.0154 EVID:4140 -- A 'Extreme failure' has occurred. Your system is been down, you...	
Group date	0-10	10/11/2008	
Group time	11-24	08:15:00.0154	
Group evid	30-34	4140	
Group msg	37-122	A 'Extreme failure' has occurred. Your system is been down, your cat has cheezburger!	

By using named groups the expression will isolate the date, time, evid and message. Character classes are used in with ranges and string literal delimiters to create each group. The message at the end of the log can be any string or a null string so '.*' is used to capture the 'msg' group.

Exercise 15

Task #1

1. Create a REGEX that will match
 1. the 4 digit EVID code in a group
 2. PROTIPS
 1. The date can change
 2. The time can change
 3. The EVID can change
 4. Don't use literals for what can change
2. Select the whole TEST STRING as a single match

Exercise 14, task #1 satisfies the answer the current task requirements.