

localhost

regex101.com

REGULAR EXPRESSION

2 matches (23 steps, 60µs)

TEST STRING

last semester i took EECS 280 and now I'm
in EECS 398

colored green because capturing group for re.findall()

(relevant)

? < 11.1 >

1:48 – match 2

The screenshot shows a browser window with two tabs: 'localhost' and 'regex101.com'. The 'regex101.com' tab displays a regular expression editor. The 'REGULAR EXPRESSION' field contains the pattern ': / EECS・(280|398) / gm'. The 'TEST STRING' field contains the sentence 'last semester i took EECS 280 and now I'm in EECS 398'. Two matches are found: 'EECS 280' and 'EECS 398'. The word 'EECS' and the digit '280' in the first match, and 'EECS' and '398' in the second, are highlighted with blue boxes, while the digits '280' and '398' are highlighted with green boxes. Handwritten red arrows point from the green boxes to the text 'colored green because capturing group for re.findall()'. Another handwritten note '(relevant)' is placed next to the first match. The status bar at the bottom right shows '1:48 – match 2'.

Example (closure, parentheses):

- What does `eeecs*` match?
- What does `(eecs)*` match?

localhost regex101.com

regex₁₀₁

REGULAR EXPRESSION 2 matches (12 steps, 55µs)

: / eeecs* / gm

TEST STRING

i·am·in·eeccssssssssssssssss·1234
and·my·friend·is·in·eecc·15

0 or more of the previous character

2:27

Example (character classes, at least one):
[A-E]+ is just shortform for (A|B|C|D|E)
(A|B|C|D|E)*.

localhost regex101.com

regular expression 6 matches (12 steps, 55μs)

:/ [A-E2-7x-z+] + / gm

TEST STRING

ABRAC234532453ADARBRA91445x2y2X+

special characters
are usually
escaped by default in
[square brackets]

1:33

The screenshot shows a browser window with two tabs: 'localhost' and 'regex101.com'. The 'regex101.com' tab displays a regular expression demo. The regular expression input field contains ':/ [A-E2-7x-z+] +' with a blue arrow pointing to the character class '[A-E2-7x-z+]' and a red arrow pointing to the plus sign '+'. To the right of the input field are buttons for 'gm' and a copy icon. Below the input field is a 'TEST STRING' input field containing the string 'ABRAC234532453ADARBRA91445x2y2X+'. Handwritten red text is overlaid on the page, explaining that special characters like '-' and '.' are usually escaped in square brackets, and that some characters like '2' and '3' are treated as literals. The bottom right corner shows the time '1:33'.

Example (number of occurrences): What does tri{3, 5} match? Does it match 'triiiii'?

localhost regex101.com

REGULAR EXPRESSION 3 matches (8 steps, 35µs)

: / [0-9]{3,5} / gm

TEST STRING

ABRAC234532453ADARBRA91445x2y2X+-

matches 3-5 consecutive digits

regex is GREEDY:
finds the longest possible matches.

1:34

escape character	<code>umich\.edu</code>	'umich.edu'	'umich!edu'
beginning of line	<code>^ark</code>	'ark two' 'ark o ark'	'dark'
end of line	<code>ark\$</code>	'dark' 'ark o ark'	'ark two'
zero or one	<code>cat?</code>	'ca' 'cat'	'cart' (matches 'ca' only)
built-in character classes*	<code>\w+</code> <code>\d+</code>	'billy' '231231'	'this person' '858 people'
character class negation	<code>[^a-z]+</code>	'WOLVERINE551' '1721\$\$'	'porch' 'billy.edu'

*Note: in Python's implementation of regex,

, [0-9]

underscores included!

- `\d` refers to digits.
- `\w` refers to alphanumeric characters ([A-Z] [a-z] [0-9] _). Whenever we say "alphanumeric" in an assignment, we're referring to `\w`!

- `\s` refers to whitespace.

- `\b` is a word boundary.

REGULAR EXPRESSION

: / insert your regular expression here

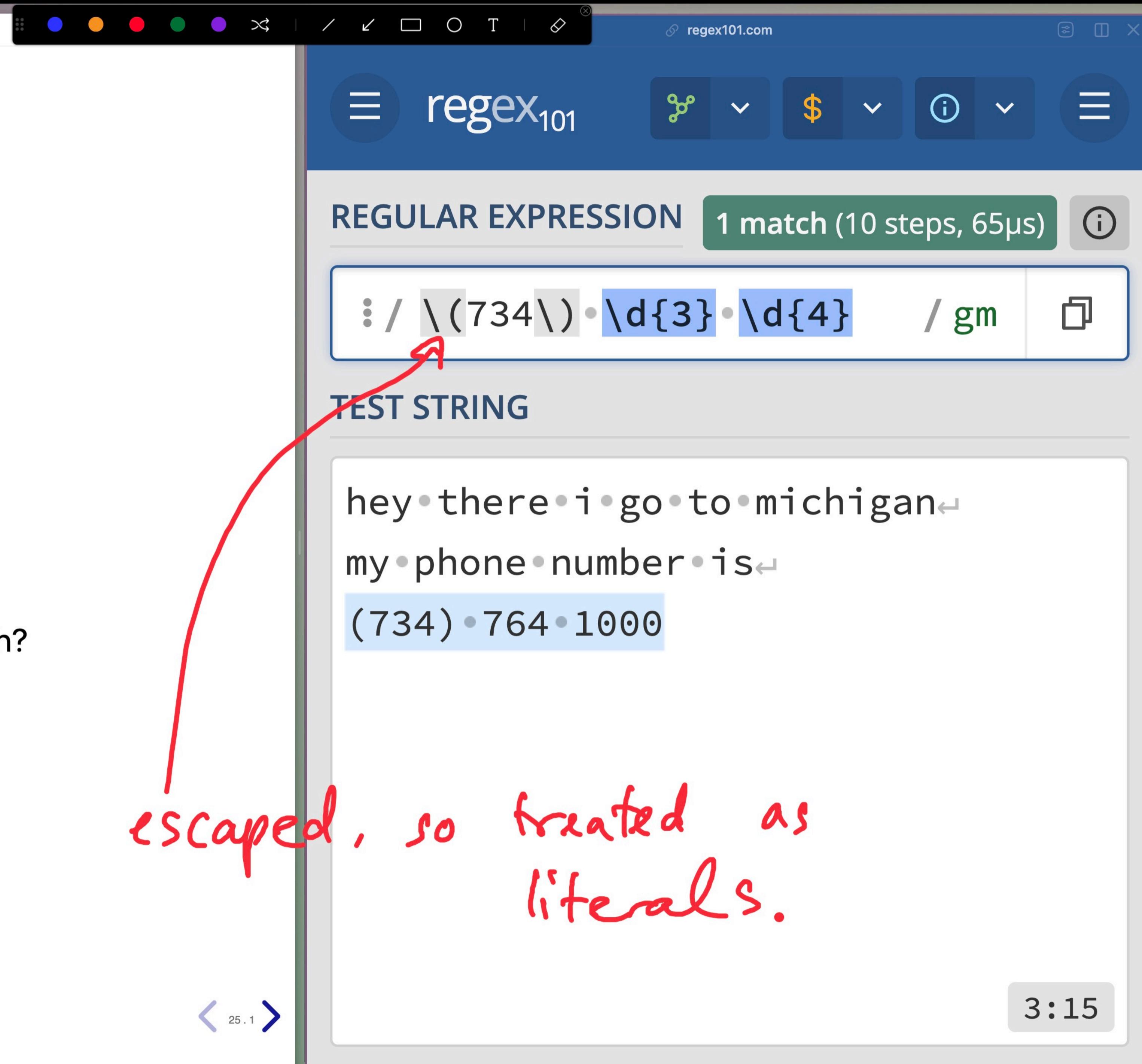
TEST STRING

insert your test string here

localhost

regex101.com

24.



- What does he. match?
 - What does he\ . match?
 - What does (734) match?
 - What does \ (734\) match?

I escaped, so treated as
literals.



Raw strings

regex \b : word boundary
Python \b : backspace

When using regular expressions in Python, it's a good idea to use **raw strings**, denoted by an `r` before the quotes, e.g. `r'exp'`.

```
In [7]: 1 re.findall('\bcat\b', 'my cat is hungry')
```

```
Out[7]: []
```

```
In [8]: 1 re.findall(r'\bcat\b', 'my cat is hungry')
```

```
Out[8]: ['cat']
```

```
In [9]: 1 # Huh?  
2 print('\bcat\b')
```

```
ca
```

raw string



```
15]: 1 # A regex that matches strings with two of the same vowel followed by  
2 # We only want to capture the digits, but...  
3 re.findall(r'(aa|ee|ii|oo|uu)(\d{3})', 'eeoo124')
```

15]: [('oo', '124')]
[I tuple with 2 elts]

To specify that we **don't** want to capture a particular group, use `:` at the start.

`?:` specifies a **non-capturing group**.

"don't capture this stuff, just match"

```
16]: 1 re.findall(r'(?:aa|ee|ii|oo|uu)(\d{3})', 'eeoo124')
```

16]: ['124']
[I string]

```
17]: 1 # A regex that matches strings with two of the same vowel followed by  
2 # We only want to capture the digits, but...  
3 re.findall(r'(aa|ee|ii|oo|uu)(\d{3})', '''  
4 check out these codes  
5  
6 eeoo124  
7 xxaa519  
8 hey there ee500  
9 ''')
```

```
17]: [('oo', '124'), ('aa', '519'), ('ee', '500')]
```

To specify that we **don't** want to capture a particular group, use `:` at the start.

`?:` specifies a **non-capturing group**.



Out[24]: []

- We can use the Series `str.findall` method, with the regular expression above, to extract hashtags out of each tweet in `tweets['text']`.

In [28]: 1 tweets['text'].head()

```
Out[28]: 0    The Best Exercise To Lose Belly Fat In 2 weeks...
           1    RT @Philanthropy: Dozens of 'hate groups' have...
           2    Artificial intelligence can find, map poverty, ...
           3    Uber balks at rules proposed by world's busies...
           4    RT @dirtroaddiva1: #IHatePokemonGoBecause he ...
Name: text, dtype: object
```

In [26]: 1 tags = tweets['text'].str.findall(r'#(\w+)')
2 tags.head()

```
Out[26]: 0    [Exercise, LoseBellyFat, CatTV, TeenWolf]
           1    []
           2    [tech]
           3    [news]
           4    [IHatePokemonGoBecause, PokesAreJokes]
Name: text, dtype: object
```

