REGEX

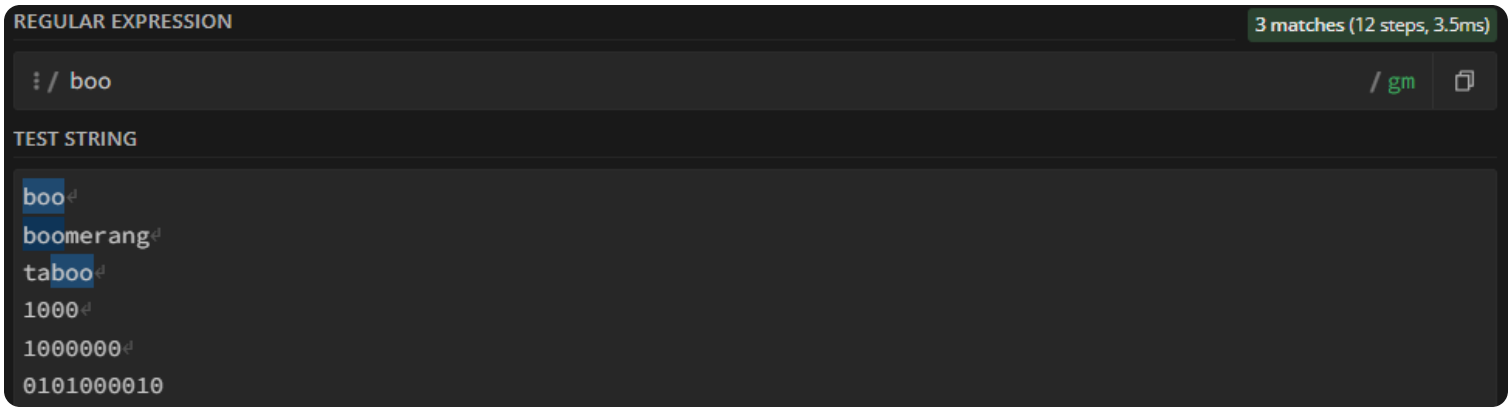
Written by Tyler Weiss 18 MAR 2024

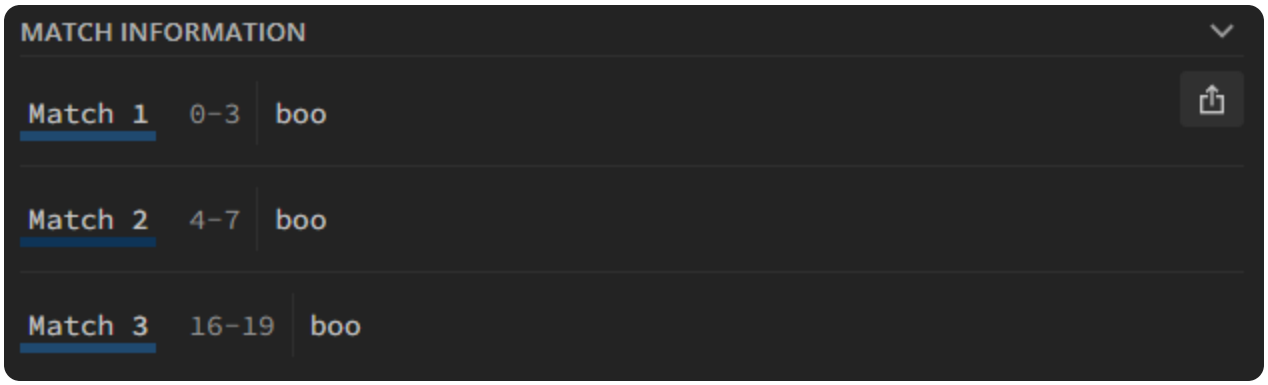
Exercise 4

Task #1

---

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



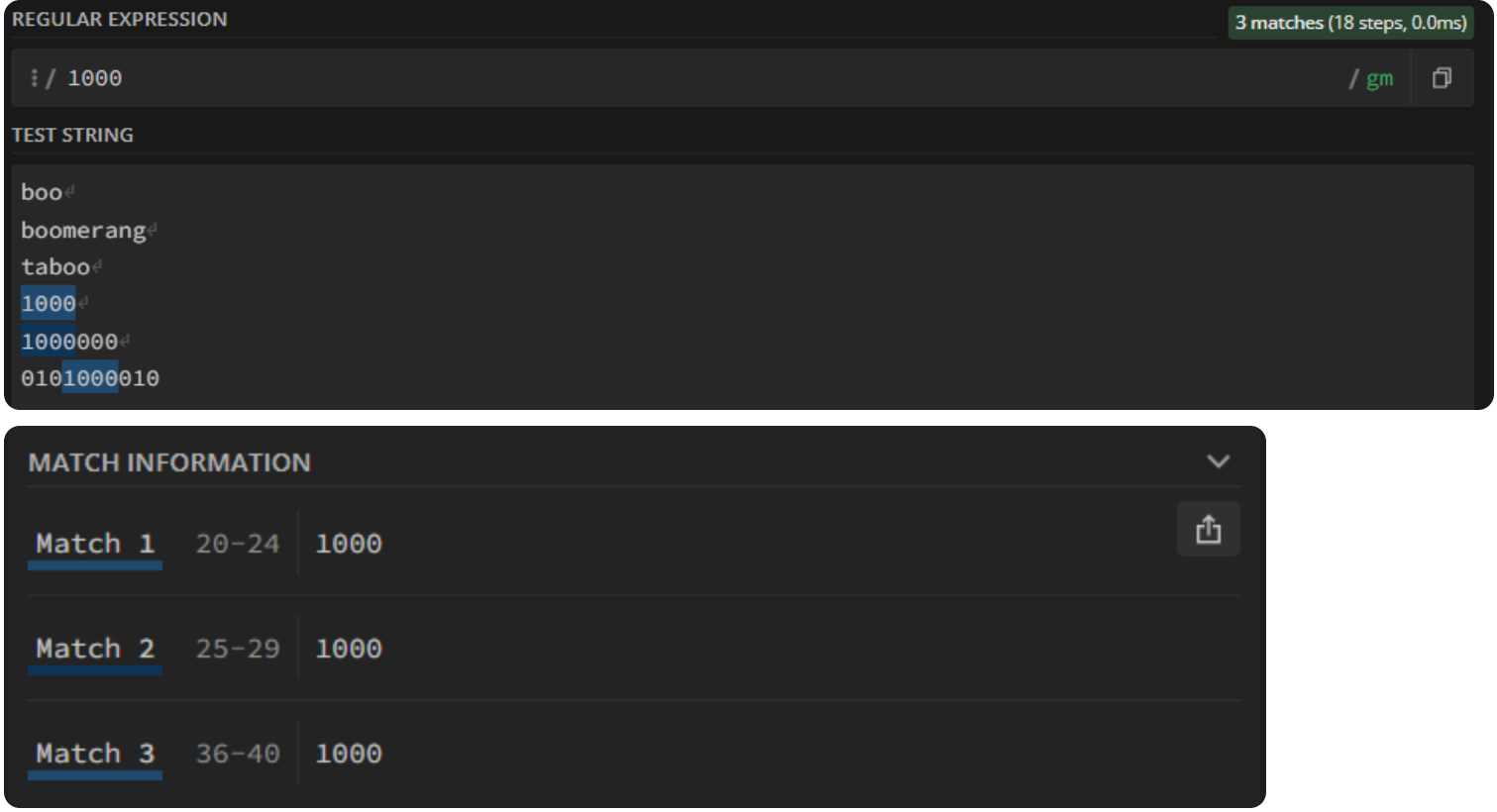


Using the literal string 'boo' as the regex argument matches any literal occurrence of the string as shown above.

Task #2

---

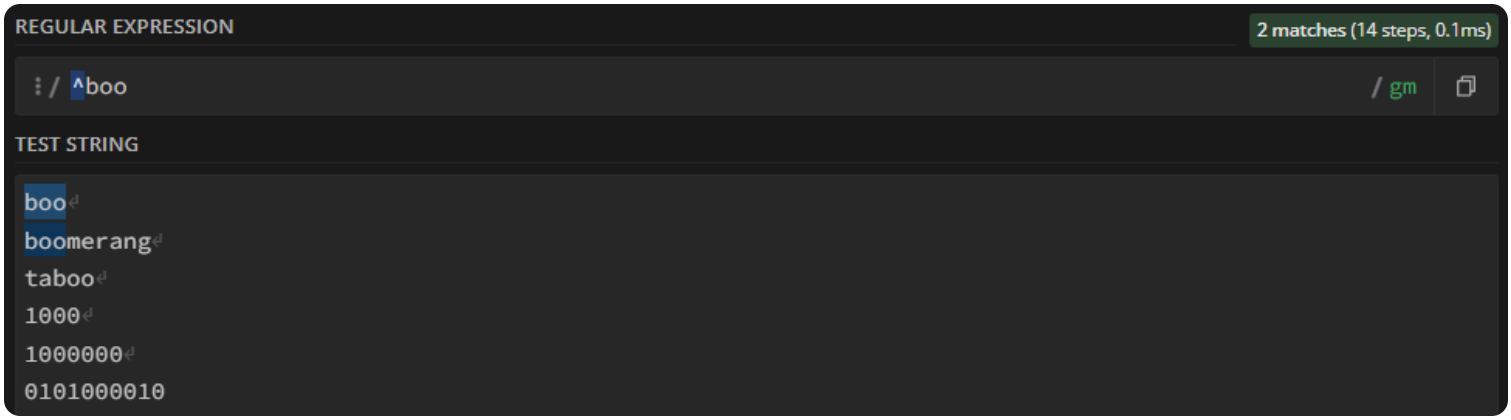
1. Use 1000 as a literal in your regex
2. Discover what it selects

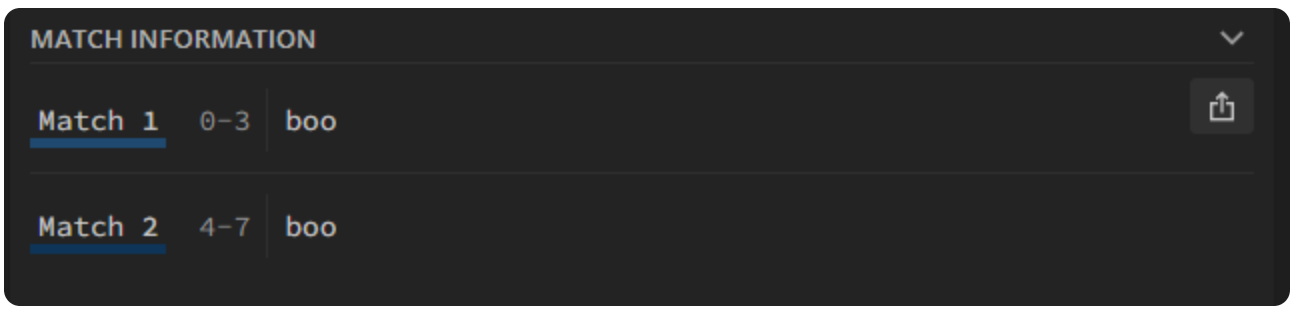


Using the literal string '1000' as the regex argument matches any literal occurrence of the string as shown above.

Task #3

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



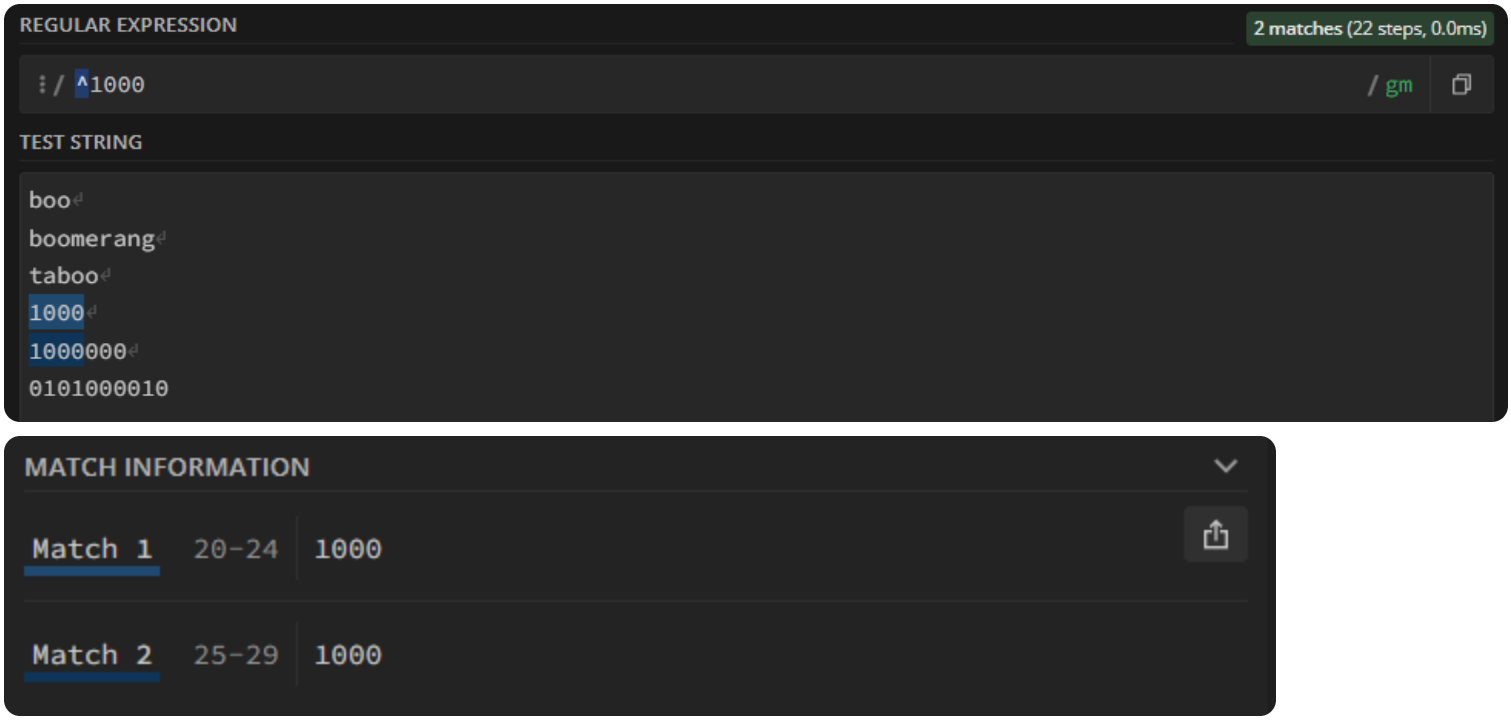


Using the anchor '^' specifies that the the regex is looking for a match at the beginning of a string.

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

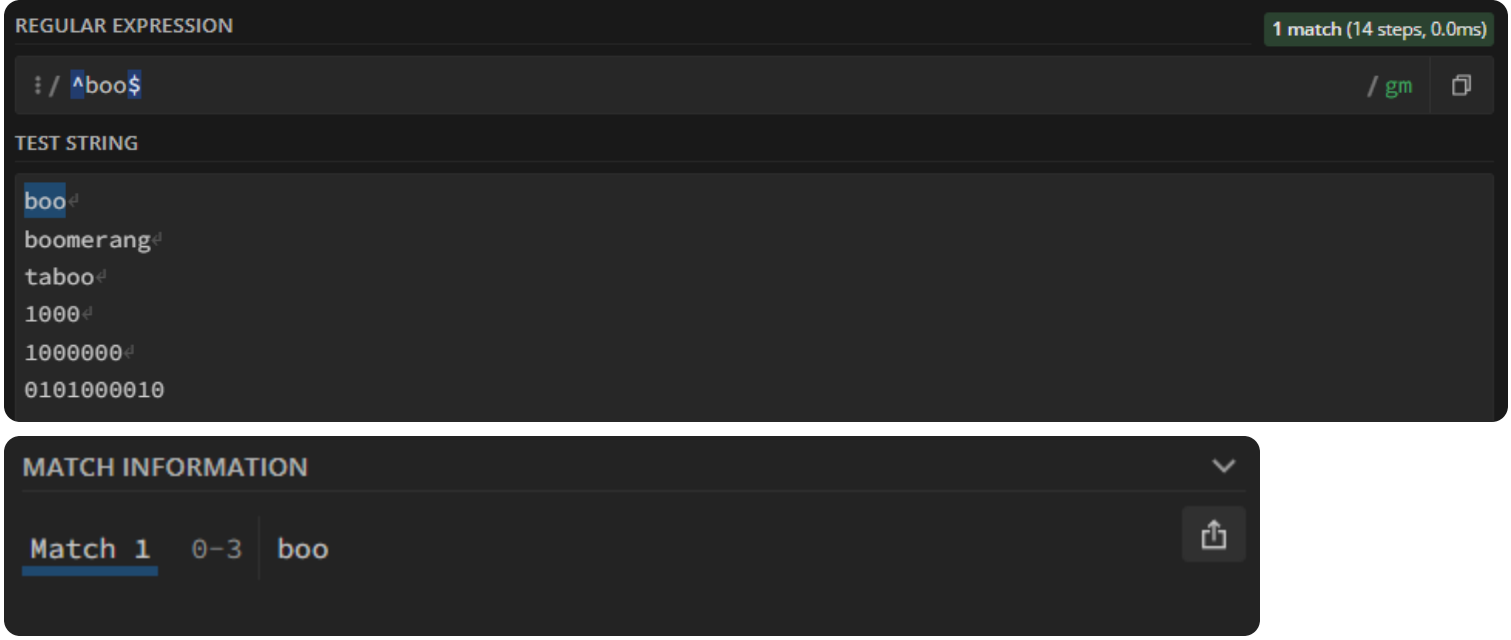
1. Use ^1000 as a literal in your regex
2. Discover what it selects



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 'boo' should only match if it occurs at the beginning of a string which resulted in the only two matches as shown above.

Task #5

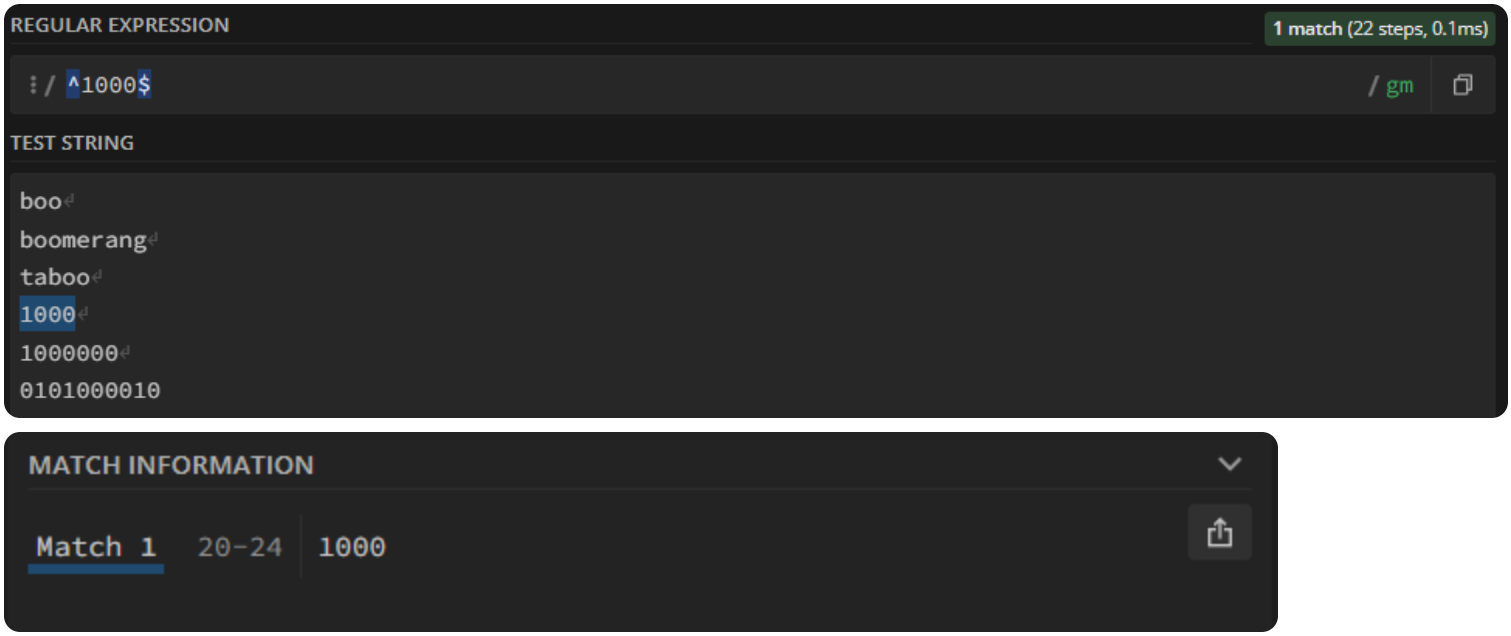
1. Use ^boo$ as a literal in your regex
2. Discover what it selects



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

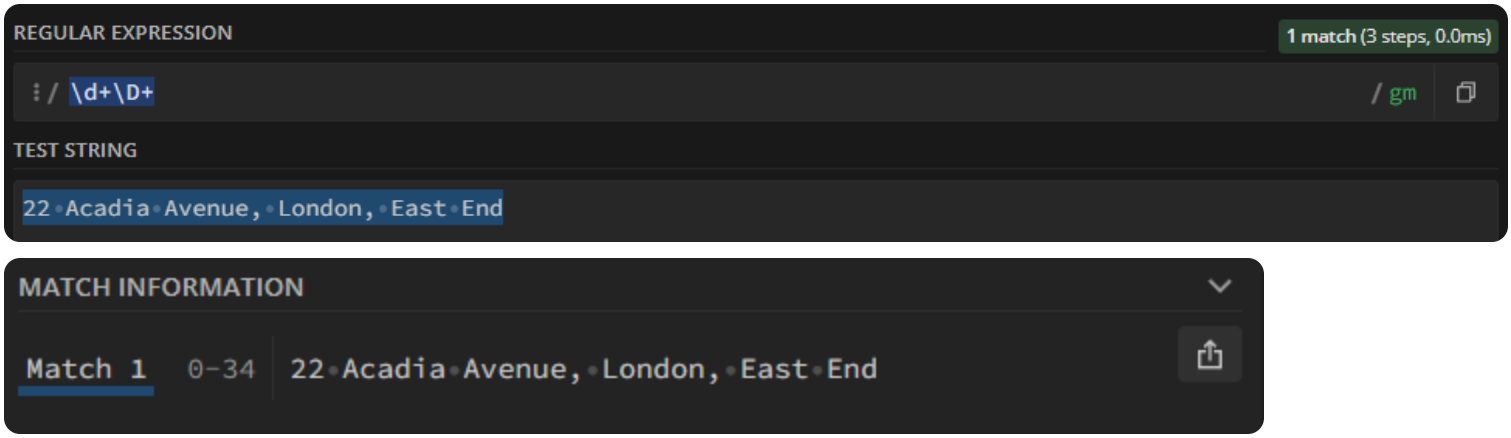


Using the anchor '^' in conjunction with the anchor '$' specifies that the regex contained from the beginning to the end of a string. The regex argument '^1000$' states that the string literal '1000' should only match if it is at the beginning and end of the string at the same time. Thus the only match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

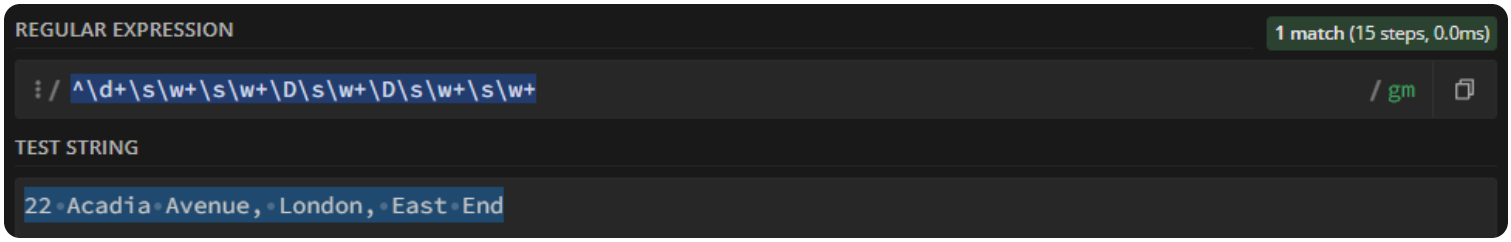
Exercise 5

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.



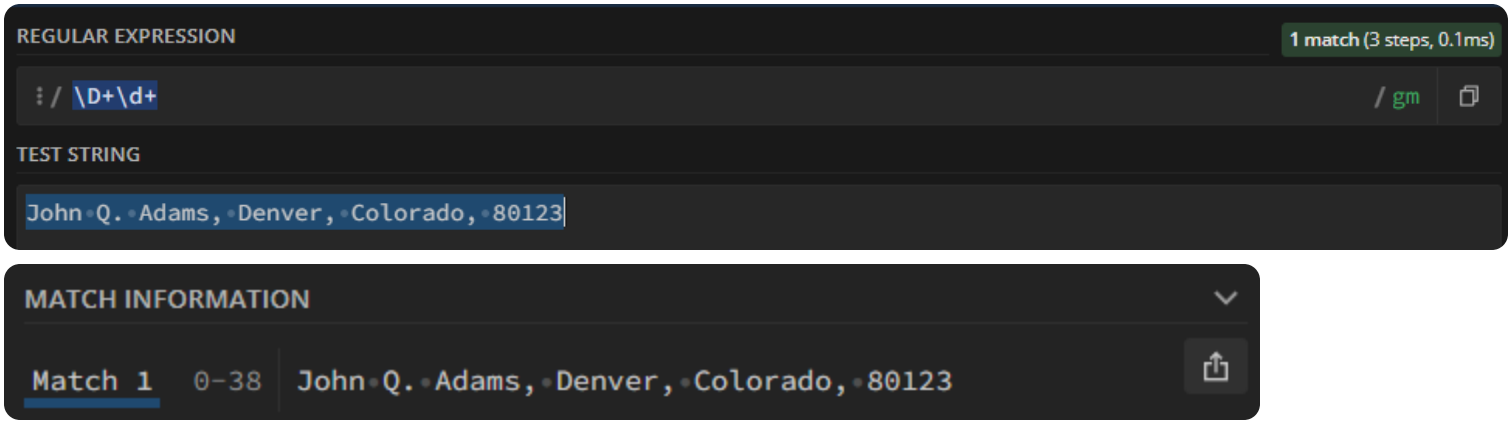
Using only the options listed above, the quickest way to match the the entire string would is to use the class '\d' with the quantifier '+' to match one or more digits. Then use the class '\D' with the quantifier '+' to match one or more characters that are non-digits. The above capture shows that this 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.



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.

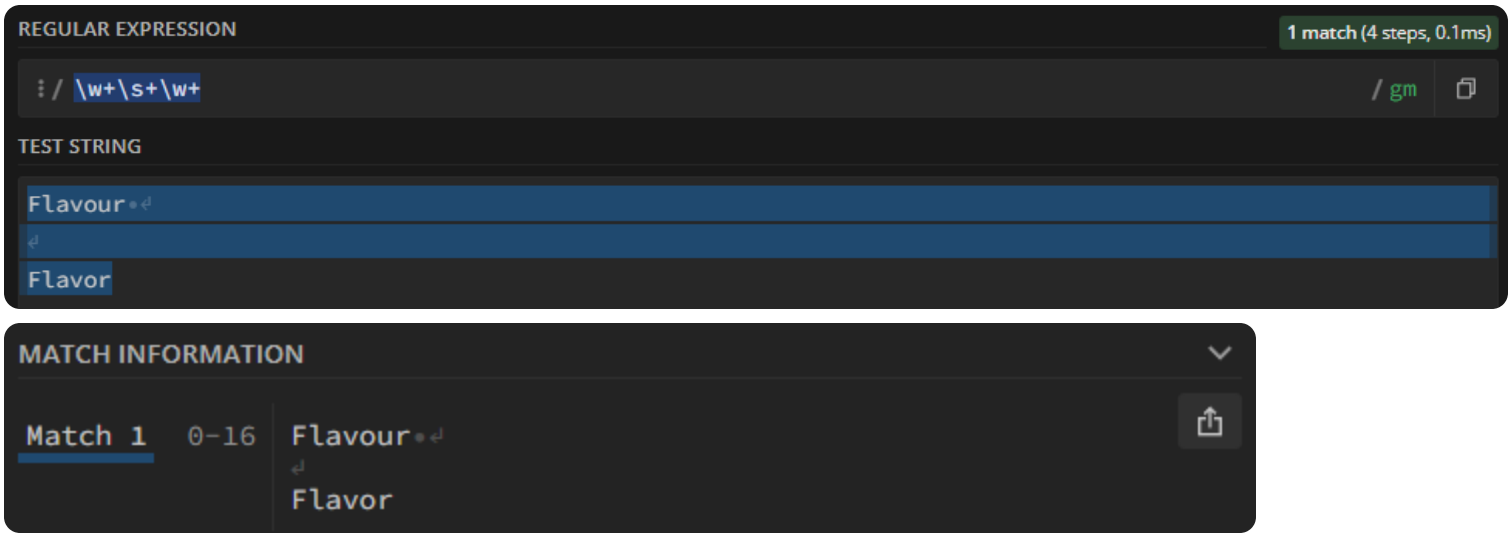


Using the class '\D' with '+' matches all non-digit characters one or more times to include spaces and commas. The class '\d' with '+' matches one or digits creating a full match for the string in the quickest possible way.

Exercise 7

Task #1

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

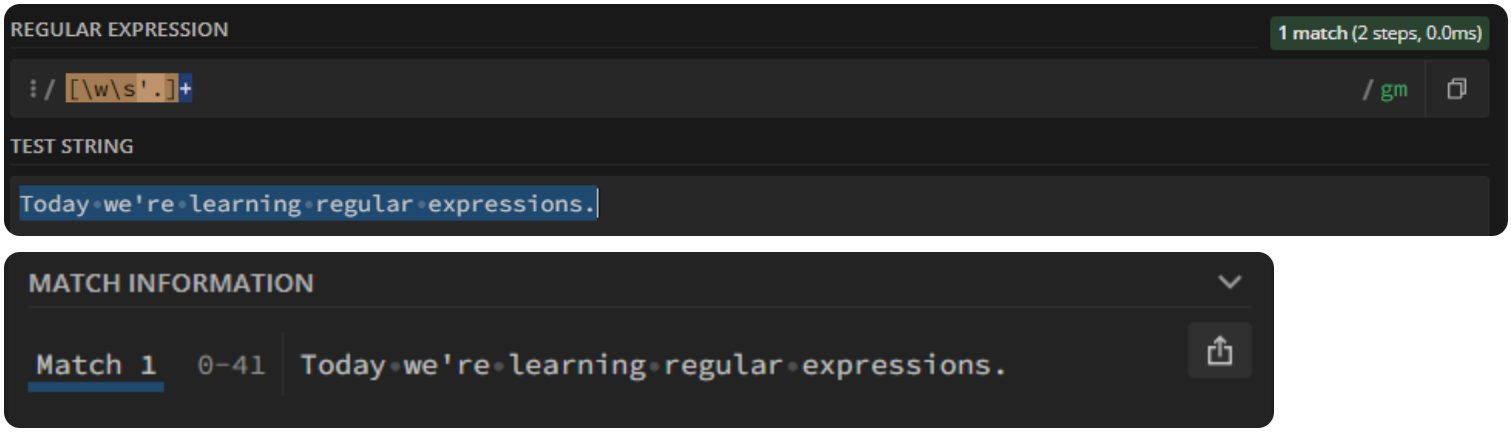


The above capture uses the class '\w' and '\s' with the quantifier appended to match the first and last words of any length with one or more space characters in between to include carriage returns.

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

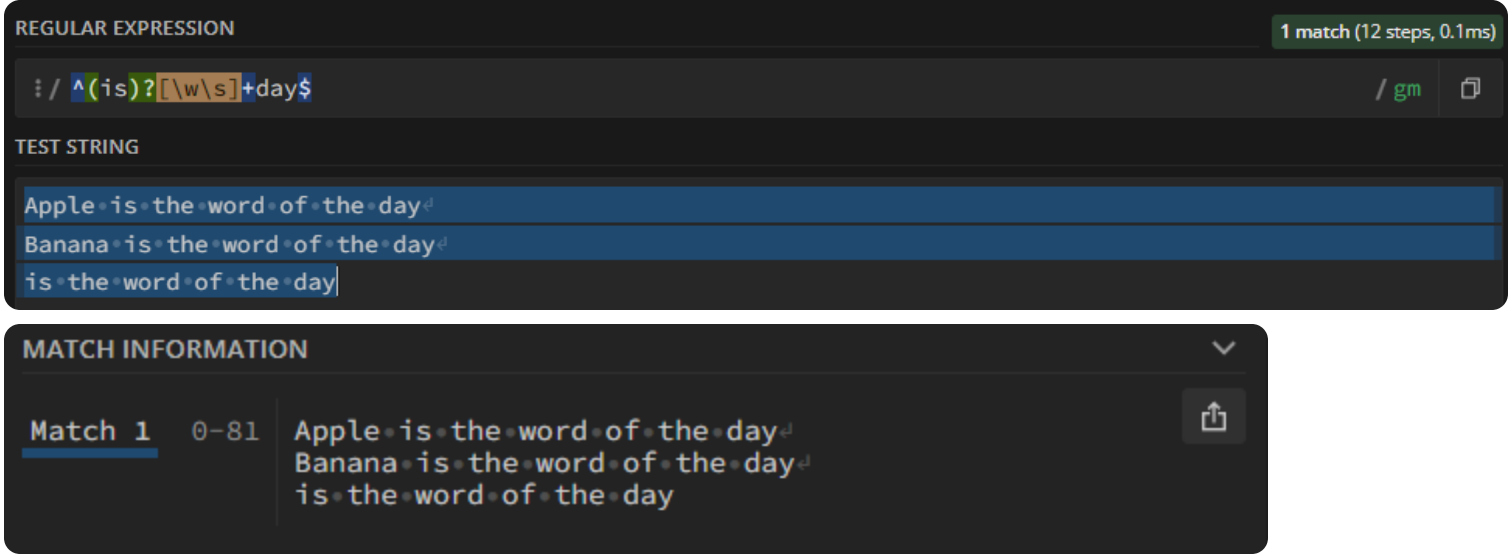


The character class used includes all numbers, letters, commas and apostrophes with the quantifier '+' to match one or more characters resulting in the selection of the entire string.

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

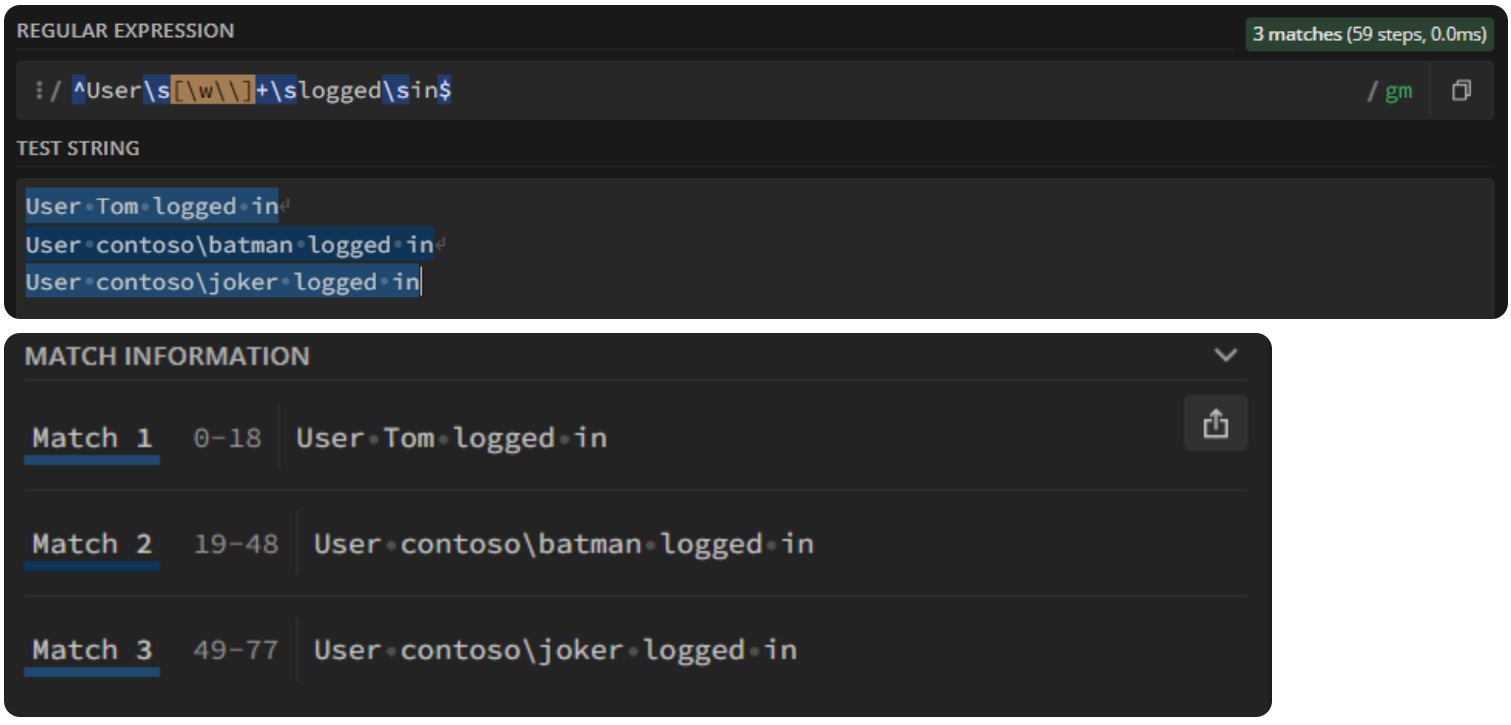


The above regex will select all three strings with 'is' being optional at the beginning of the string and 'day' at the end. This expression will also select all three strings as a single match.

Exercise 10

Task #1

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



Exercise 11

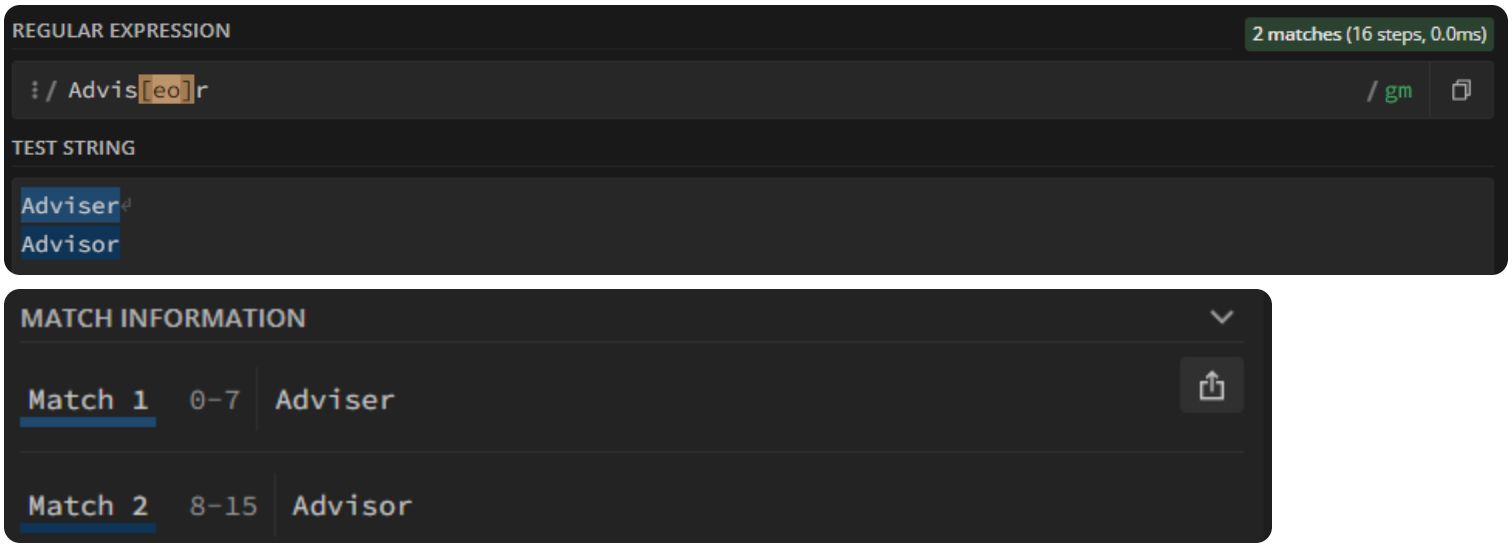
Task #1

1. Match both strings
2. Must use a class

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

Task #2

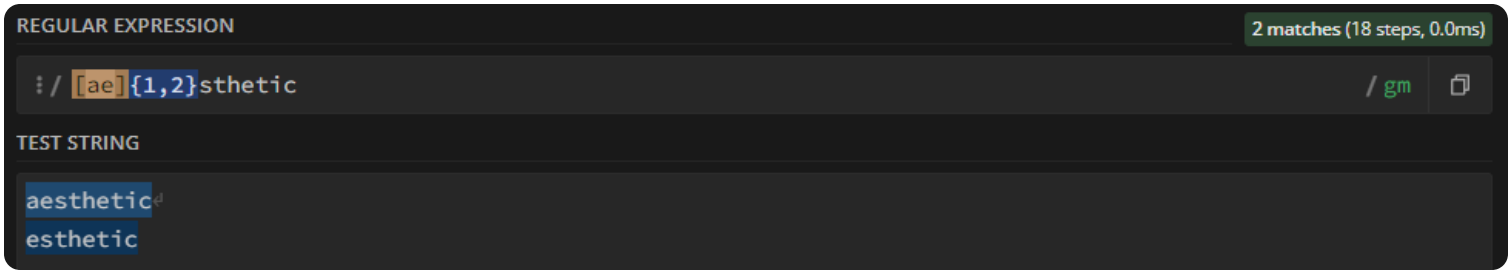
1. Match both strings
2. Must use a class

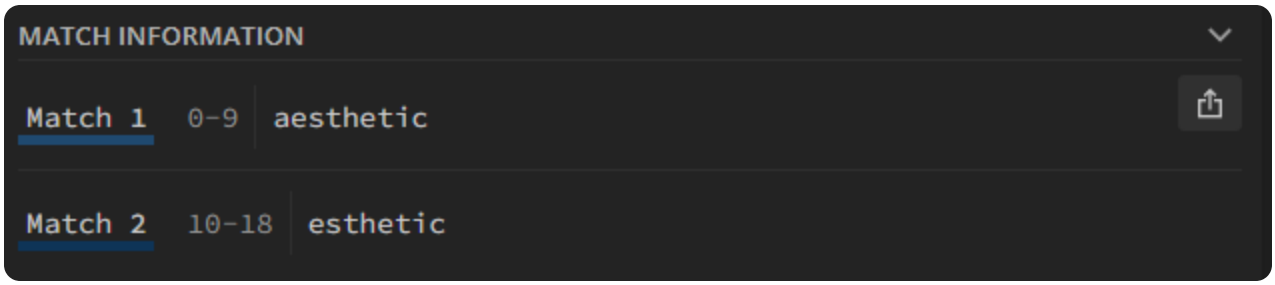


Using a the class '[eo]' the expression can select both spellings of the strings.

Task #3

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

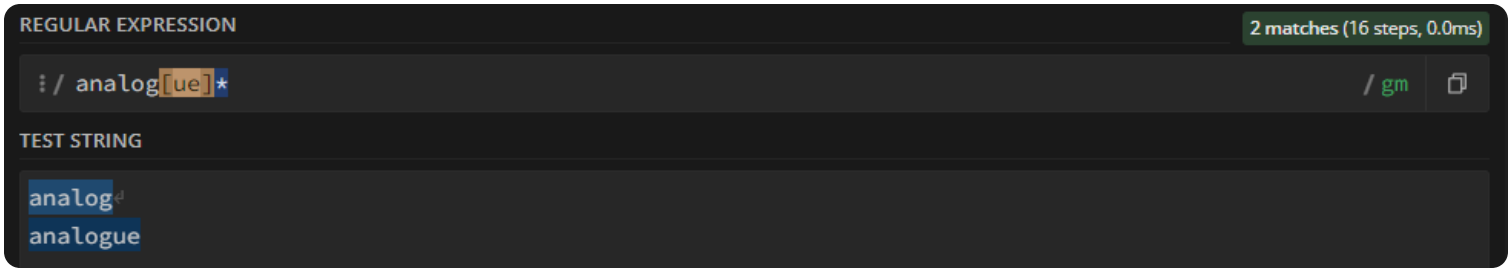


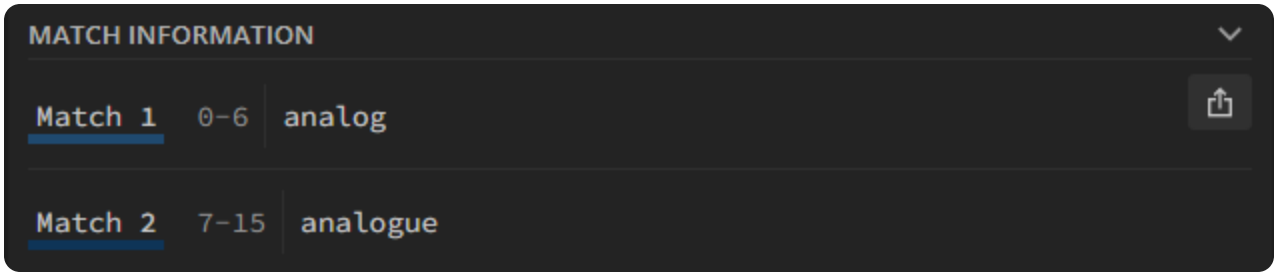


Using a the class '[ae]' in conjunction with a range of one to two characters the expression can select both spellings of the strings.

Task #4

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



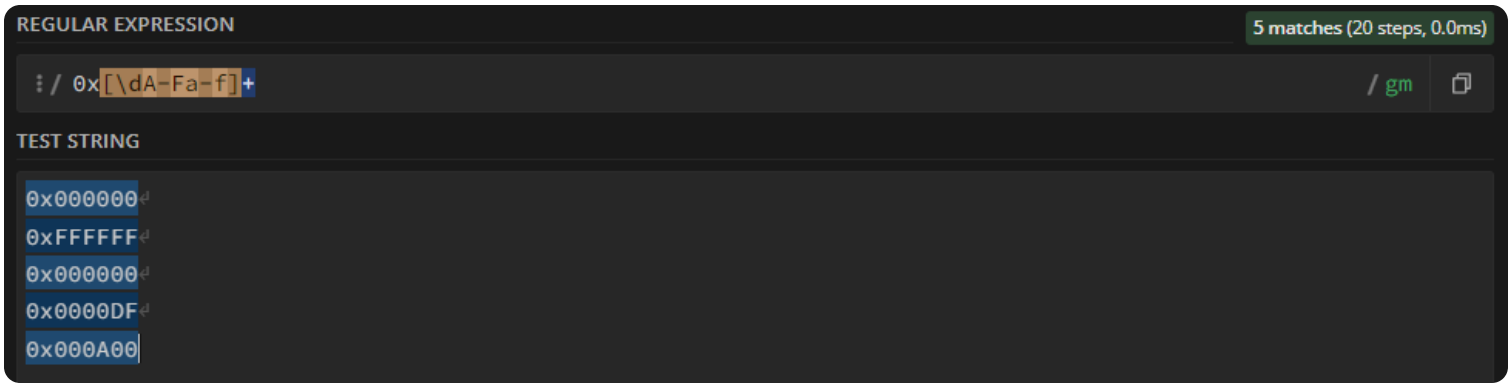


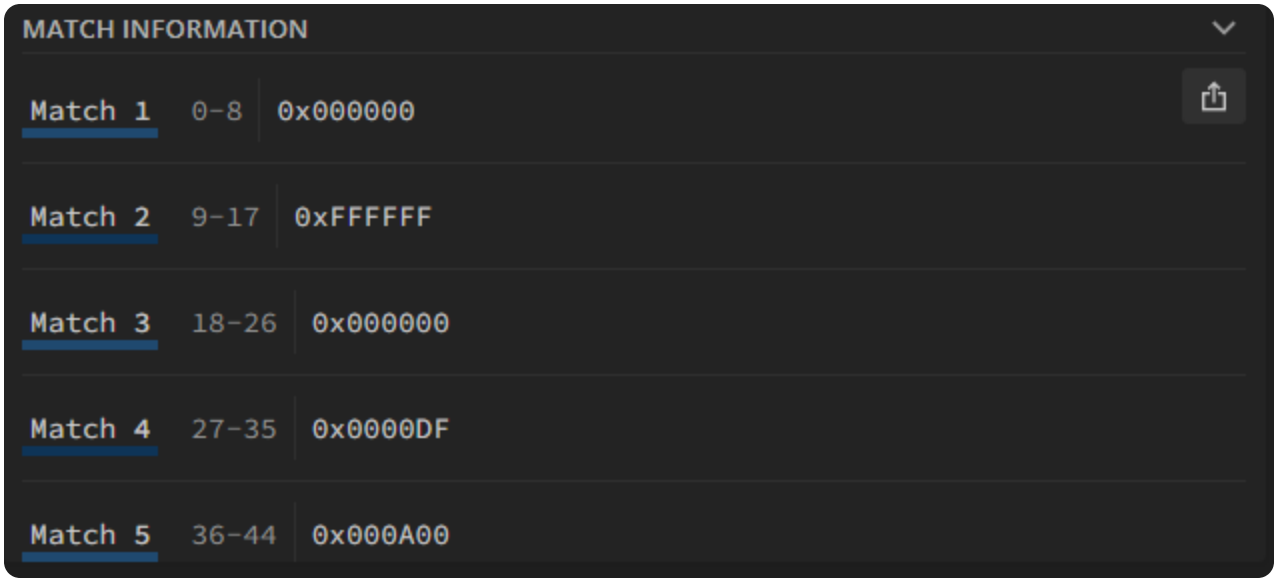
Using a the class '[ue]' in conjunction with the quantifier '*' to select zero or more characters, the expression can select both spellings of the strings.

Exercise 12

Task #1

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





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

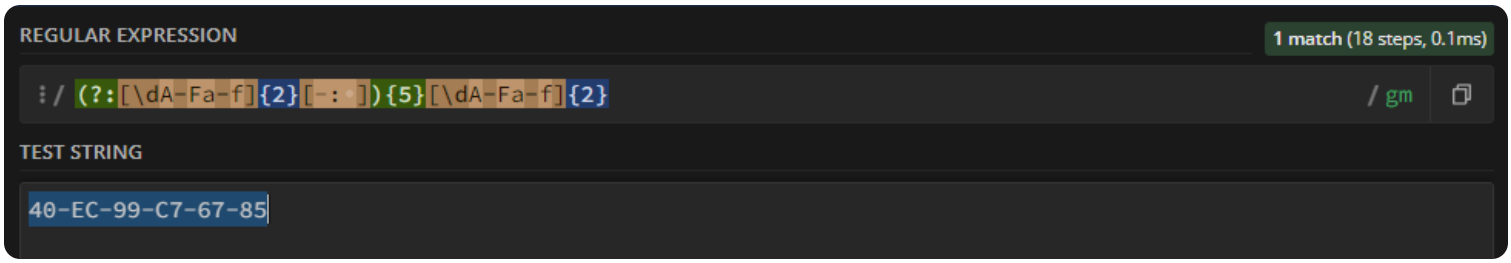
Task #1

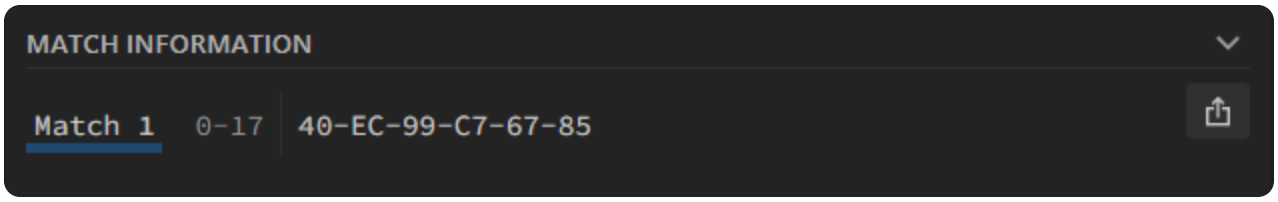
1. Match any MAC address
2. Must use a classes built of base-16 ranges (hexadecimal)

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
```

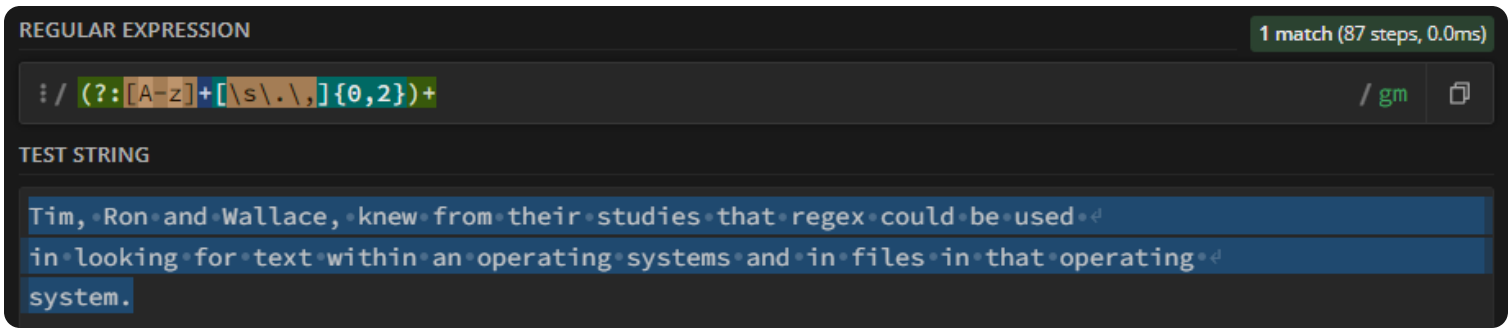


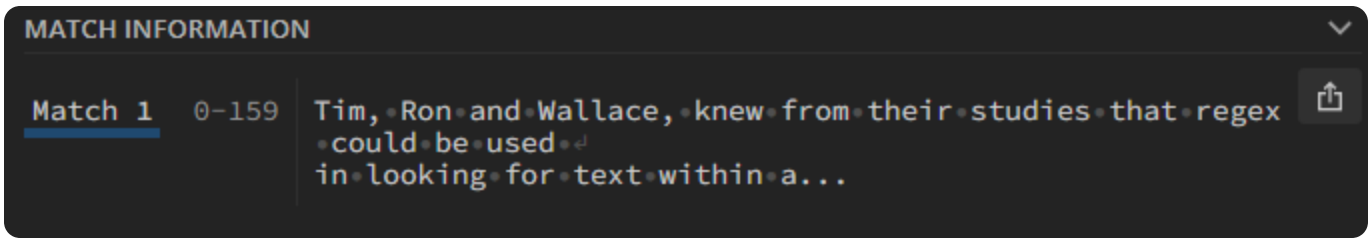


Using the class '[\dA-Fa-f]' will capture any hex character. The class '[-:]' will capture all characters that separate the MAC address octets. So by using a non-capture group that specifies an octet followed by the possible delimiters we can match the first 5 octets with their delimiters. Then the last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



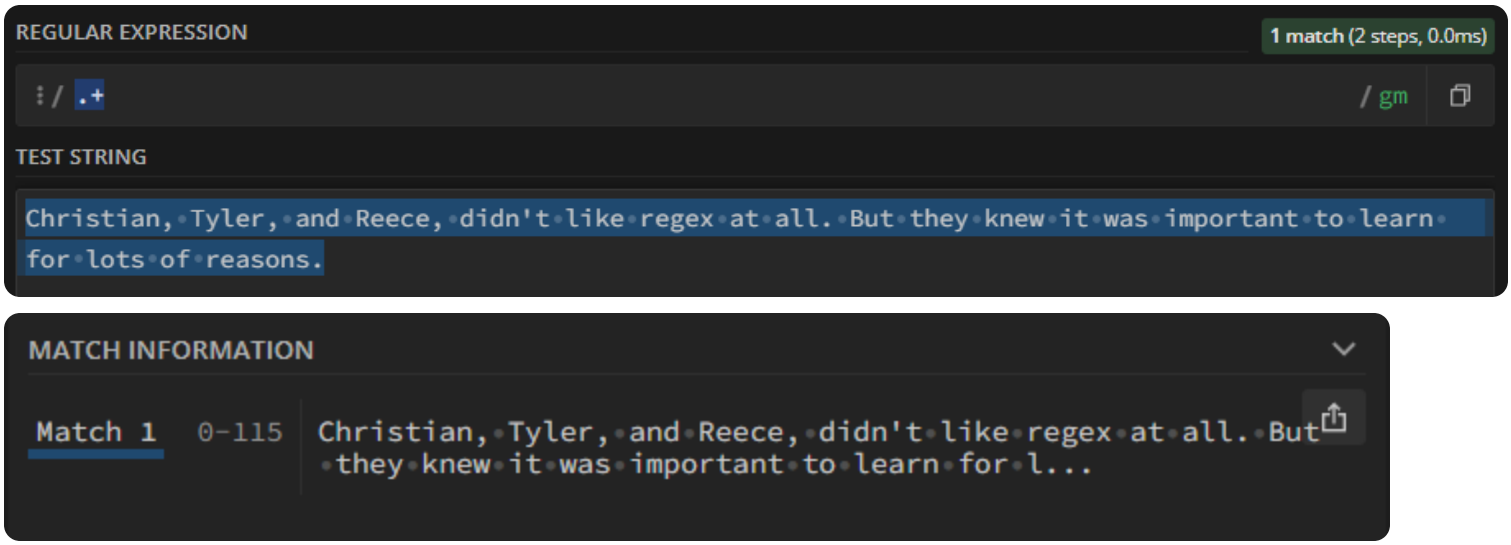


The class '[A-z]' with the quantifier '+' will select the words in the string. Adding the class '[\s.,]' for a range of '{0,2}' will also select the space and punctuation characters after each word. With the

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

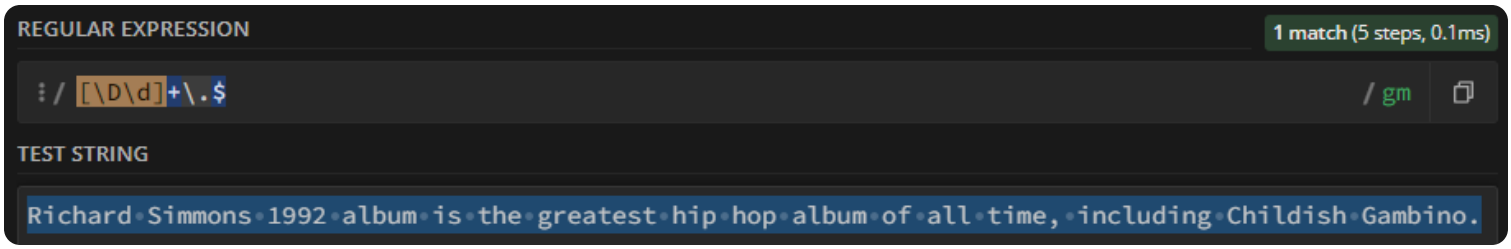
1. Only using 2 characters in the expression, select the whole string

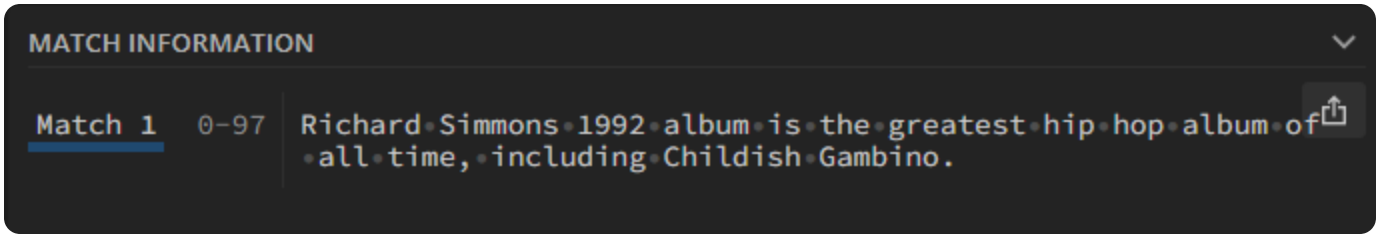


Only using 2 characters in the expression we have to use the '.' metacharacter which represents all characters. If the quantifier '+' is used the expression will then select all 1 or more of any character. Note, do not use the quantifier '*' because that will select zero or more characters, the string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





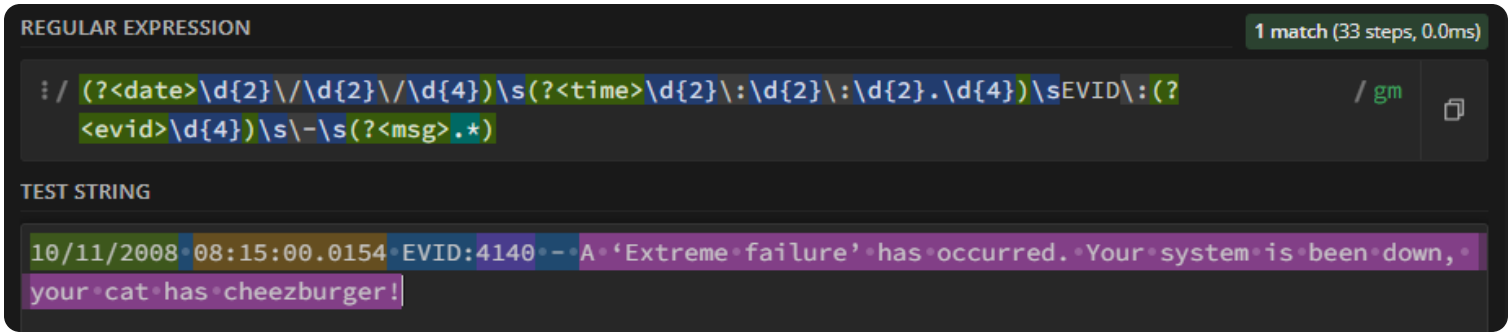
By using the class '[\D\d]' with the quantifier '+' the expression will select all non-digit characters and

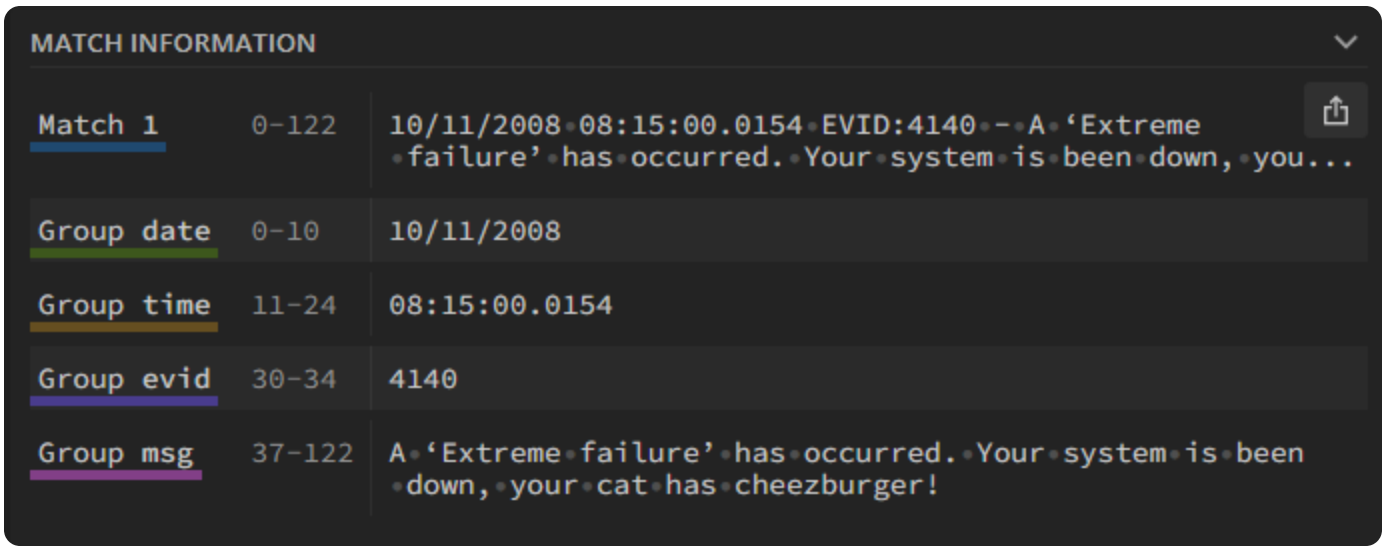
all digits 1 or more times. The '\.$' anchor was added to the end creating a delimiter, thus the expression would select entire sentences if they end in a period.

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





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.