========================================

**Regex grouping using OR "|"**

========================================

1) or is represented using pipe symbol i.e |

2) You can define group using parenthesis ()

3) You can define explicit character ranges using [ ]

So, for example if you want to find numbers as well as words

in sentence following will be the syntax :

re.findall('(\d+|\w+)', " Aditya turned 26 in 2020.")

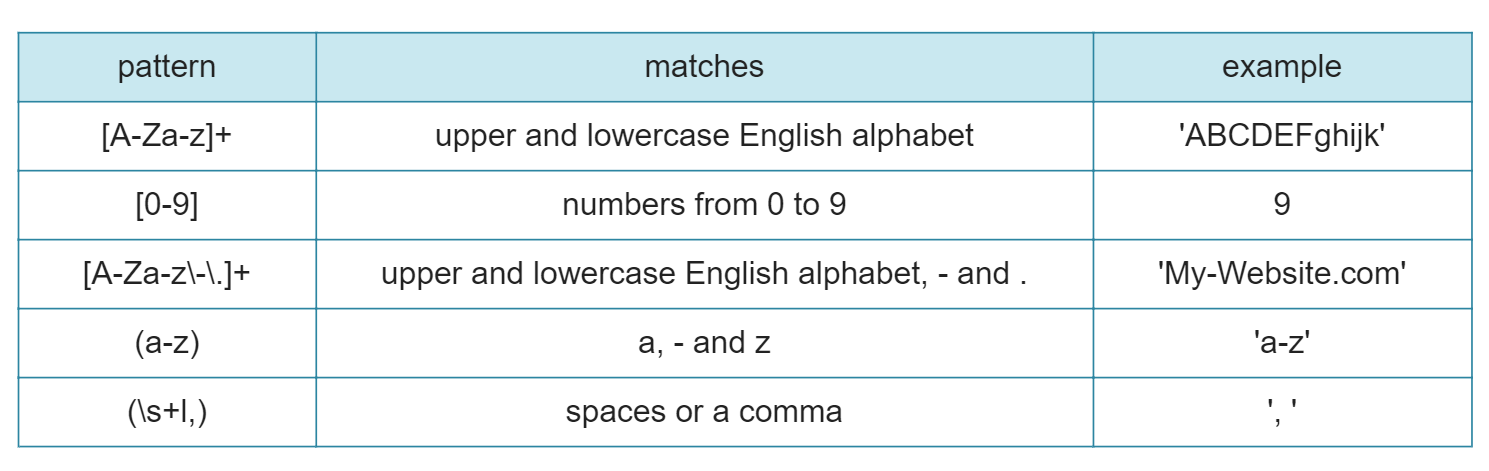
output == > ['Aditya', 'turned', '26', 'in', '2020']

NOTE: punctuation is not counted-in in this operation

====================================

**Regex ranges and groups**

====================================



====================================

**Character ranges with re.match()**

====================================

my\_str = 'match lowercase spaces nums like 12, but no commas'

re.match('[a-z0-9 ]+', my\_str)

output 🡺 match='match lowercase spaces nums like 12'>

So, in above code we are doing greedy search. As soon as we hit , (comma) we are out.

Remember + (plus symbol) is used to do greedy search.

Twitter is a frequently used source for NLP text and tasks. In this exercise, you'll build a more complex tokenizer for tweets with hashtags and mentions using nltk and regex.

Exercises: [ you can run thin in jupyter or spyder IDE. Following are questions with answers. You can try yourself to brush your skills that you have learned so far]

Tweets = ['This is the best #nlp exercise ive found online! #python', '#NLP is super fun! <3 #learning', 'Thanks @datacamp :) #nlp #python']

**#1. Define a regex pattern to find hashtags: pattern1, Use the pattern on the first tweet in the tweets list**

* # Import the necessary modules

from nltk.tokenize import regexp\_tokenize

from nltk.tokenize import TweetTokenizer

# Define a regex pattern to find hashtags: pattern1

pattern1 = r"#\w+"

# Use the pattern on the first tweet in the tweets list

hashtags = regexp\_tokenize(tweets[0], pattern1)

* print(hashtags)

**#2. Write a pattern that matches both mentions (@) and hashtags. Use the pattern on the last tweet in the tweets list**

# Import the necessary modules

from nltk.tokenize import regexp\_tokenize

from nltk.tokenize import TweetTokenizer

# Write a pattern that matches both mentions (@) and hashtags

pattern2 = r"([@#]\w+)"

# Use the pattern on the last tweet in the tweets list

mentions\_hashtags = regexp\_tokenize(tweets[-1],pattern2)

print(mentions\_hashtags)

**#3. Use the TweetTokenizer to tokenize all tweets into one list**

# Import the necessary modules

from nltk.tokenize import regexp\_tokenize

from nltk.tokenize import TweetTokenizer

# Use the TweetTokenizer to tokenize all tweets into one list

tknzr = TweetTokenizer()

all\_tokens = [tknzr.tokenize(t) for t in tweets]

print(all\_tokens)

**Non-ascii tokenization**

**In this exercise, you'll practice advanced tokenization by tokenizing some non-ascii based text. You'll be using German with emoji!**

german\_text = Wann gehen wir Pizza essen? 🍕 Und fährst du mit Über? 🚕

**#1. Tokenize and print all words in german\_text**

all\_words = word\_tokenize(german\_text)

print(all\_words)

**# Tokenize and print only capital words**

capital\_words = r"[A-Z|Ü]\w+"

print(regexp\_tokenize(german\_text, capital\_words))

**# Tokenize and print only emoji**

emoji = "['\U0001F300-\U0001F5FF'|'\U0001F600-\U0001F64F'|'\U0001F680-\U0001F6FF'|'\u2600-\u26FF\u2700-\u27BF']"

print(regexp\_tokenize(german\_text,emoji))