Video 36 : Regular Expressions 6

In this video I finish my Regular Expressions coverage. We'll look at Or, Group, Named Groups, More Match Object Functions and then we'll solve some problems.

Overview of What We've Learned About Regex

```
# [ ]   : Match what is in the brackets
# [^ ]  : Match anything not in the brackets
# ( )   : Return surrounded submatch
# .     : Match any 1 character or space
# +     : Match 1 or more of what proceeds
# ?     : Match 0 or 1
# *     : Match 0 or More
# *?    : Lazy match the smallest match
# \b    : Word boundary
# ^     : Beginning of String
# $     : End of String
# \n    : Newline
# \d    : Any 1 number
# \D    : Anything but a number
# \w    : Same as [a-zA-Z0-9_]
# \W    : Same as [^a-zA-Z0-9_]
# \s    : Same as [\f\n\r\t\v]
# \S    : Same as [^\f\n\r\t\v]
# {5}   : Match 5 of what proceeds the curly brackets
# {5,7} : Match values that are between 5 and 7 in length
# ($m)  : Allow ^ on multiline string

# Use a back reference to substitute what is between the
# bold tags and eliminate the bold tags
# re.sub(r"<b>(.*?)</b>", r"\1", randStr)

# Use a look ahead to find all characters of 1 or more
# with a word boundary, but don't return the word
# boundary
# re.findall(r"\w+(?=\b)", randStr)

# Use a look behind to find words starting with a number,
# period and space, but only return the word that follows
# re.findall(r"(?<=\d.\s)\w+", randStr)

# Use a negative look behind to only return numbers without
# a $ in front of them
# re.findall(r"(?<!\$)\d+", randStr)
```

Or Conditional

You can use | to define the matches you'll except

```
rand_str = "1. Dog 2. Cat 3. Turtle"

regex = re.compile(r"\d\.\s(Dog|Cat)")
```

```
matches = re.findall(regex, rand_str)

print(len(matches))

for i in matches:
    print(i)
```

Python Problem for you to Solve

Create a regex that will match for 5 digit zip codes or zip codes with 5 digits a dash and then 4 digits. Here is sample data :

```
rand_str = "12345 12345-1234 1234 12346-333"
```

Solution

```
rand_str = "12345 12345-1234 1234 12346-333”
regex = re.compile(r"(\d{5}-\d{4}|\d{5}\s)")

matches = re.findall(regex, rand_str)

print(len(matches))

for i in matches:
    print(i)
```

Group

We can use group to retrieve parts of regex matches

```
bd = input("Enter your birthday (mm-dd-yyyy) : ")

bd_regex = re.search(r"(\d{1,2})-(\d{1,2})-(\d{4})", bd)

print("You were born on", bd_regex.group())
print("Birth Month", bd_regex.group(1))
print("Birth Day", bd_regex.group(2))
print("Birth Year", bd_regex.group(3))
```

Match Object Functions

There are functions that provide more information on your matches

```
match = re.search(r"\d{2}", "The chicken weighed 13 lbs")

# Print the match
print("Match :", match.group())

# Print the start and ending index of the match
print("Span :", match.span())

# Print starting index of the match
```

```
print("Match :", match.start())

# Print the ending index of the match
print("Match :", match.end())
```

Named Groups

You can also assign names to matches.

```
rand_str = "December 21 1974"

regex = r"^(?P<month>\w+)\s(?P<day>\d+)\s(?P<year>\d+)"

matches = re.search(regex, rand_str)

print("Month :", matches.group('month'))
print("Day :", matches.group('day'))
print("Year :", matches.group('year'))
```

Python Problem for you to Solve

Find all of the following real email addresses in this sample data.

```
rand_str = "d+b@aol.com a_1@yahoo.co.uk A-100@m-b.INTERNATIONAL"
```

Solution

```
rand_str = "d+b@aol.com a_1@yahoo.co.uk A-100@m-b.INTERNATIONAL"
regex = re.compile(r"[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+")

matches = re.findall(regex, randStr)

print(len(matches))

for i in matches:
    print(i)
```

Python Problem for you to Solve

For your final Python / Regex problem I want you to match all of the following phone numbers and then print them.

```
rand_str = "14125551212 4125551212 (412)5551212 412 555 1212 412-555-1212
1-412-555-1212”
```

Solution

```
rand_str = "14125551212 4125551212 (412)5551212 412 555 1212 412-555-1212
1-412-555-1212”
regex = re.compile(r"((1?)(-| ?)(\()?(\d{3})(\)|-| |\)-|\) )?(\d{3})(-| )?(\d{4}|\d{4}))")

matches = re.findall(regex, randStr)
```

```
print(len(matches))

for i in matches:
    print(i[0].lstrip())
```

Thank you for taking this Regex journey with me. I hope that I was able to show you how powerful Regular Expressions can be, while at the same time making them easy to understand and grasp.

In the next part of my tutorial I'll show you how to work with databases using Python.