Video 27 : Regular Expressions

Regular expressions allow you to locate and change strings in very powerful ways. They work in almost exactly the same way in every programming language as well.

Regular Expressions (Regex) are used to :

1. Search for a specific string in a large amount of data
2. Verify that a string has the proper format (Email, Phone #)
3. Find a string and replace it with another string
4. Format data into the proper form for importing for example

First we'll search for an exact string match.

CODE

```
# import the Regex module
import re

# Search for ape in the string
if re.search("ape", "The ape was at the apex"):
    print("There is an ape")
```

Get All Matches

findall() returns a list of matches and . is used to match any 1 character or space. Finditer can be used to return an iterator of matches.

CODE

```
all_apes = re.findall("ape.", "The ape was at the apex")

for i in all_apes:
    print(i)

# finditer returns an iterator of matching objects
# You can use span to get the location

the_str = "The ape was at the apex"

for i in re.finditer("ape.", the_str):

    # Span returns a tuple
    loc_tuple = i.span()

    print(loc_tuple)

    # Slice the match out using the tuple values
    print(the_str[loc_tuple[0]:loc_tuple[1]])
```

Match 1 of Several Letters

Square brackets will match any one of the characters between the brackets not including upper and lowercase varieties unless they are listed. We can also define characters in a range and define that we want to match anything except a defined number of characters.

CODE

```
animal_str = "Cat rat mat fat pat"

all_animals = re.findall("[crmfp]at", animal_str)
for i in all_animals:
    print(i)

print()

# We can also allow for characters in a range
# Remember to include upper and lowercase letters
some_animals = re.findall("[c-mC-M]at", animal_str)
for i in some_animals:
    print(i)

print()

# Use ^ to denote any character but whatever characters are
# between the brackets
some_animals = re.findall("[^Cr]at", animal_str)
for i in some_animals:
    print(i)

print()
```

Replace All Matches

You can replace items and define pattern objects.

CODE

```
# Replace matching items in a string

owl_food = "rat cat mat pat"

# You can compile a regex into pattern objects which
# provide additional methods
regex = re.compile("[cr]at")

# sub() replaces items that match the regex in the string
# with the 1st attribute string passed to sub
owl_food = regex.sub("owl", owl_food)

print(owl_food)
```

Solving Backslash Problems

Regex use the backslash to designate special characters and Python does the same inside strings which causes issues.

CODE

```
# Let's try to get "\\stuff" out of a string
rand_str = "Here is \\stuff"

# This won't find it
print("Find \\stuff : ", re.search("\\stuff", rand_str))

# This does, but we have to put in 4 slashes which is
# messy
print("Find \\stuff : ", re.search("\\\\stuff", rand_str))

# You can get around this by using raw strings which
# don't treat backslashes as special
print("Find \\stuff : ", re.search(r"\\stuff", rand_str))
```

That’s all for this video. In the next video I’ll provide much more on what you can do with regular expressions.