Course Name:- Data Cleaning in Python : Advanced

Mission Name:- Regular Expression Basics

* Example of Regular Expression:- (.+)://([\w\.]+)/?(.*)

* why to use them?

i) Save's time.
ii) Faster to execute.
iii) Used by modern programming languages and databases.

* Python has a regular expression module built-in.
Module name:- re.

* search() function of re:-

```
import re
m = re.search("and", "hand")
```

("and" → pattern, "hand" → string)

If match found, return's a match object else returns None.)

* Set:-

[msb]end

- [→ set start
- msb → any of m, s or b
-] → set end
- end → End part of substring.

* Series.str.contains() → used to test whether a series of strings match a particular regex pattern.

Eg:-
```
pattern = "[Pp]ython"
titles = hn["title"]
python_mentions = titles.str.contains(pattern).sum()
print(python_mentions)
```

PAGE:
DATE: / /

* Printing titles with word Python/python present in it :-

```
py-titles = titles[titles.str.contains("[Pp]ython")]
print(py-titles.head())
```

Quantifiers:-

Specifies the type of previous pattern character required.

chart:-

Quantifier	Pattern	Equivalent	Explanation
Numeric	a{3}	—	a, 3 times
,,	a{3,5}	—	a, 3, 4 or 5 times
,,	a{,3}	—	a, 0, 1, 2 or 3 times
,,	a{8,}	—	a, 8 or more times
Zero or more	a*	a{0,}	
One or more	a+	a{1,}	
optional	a?	a{0,1}	a, 0 or 1 time.

Eg:- email_bool = titles.str.contains("e-?mail")
email_count = email_bool.sum()
email_titles = titles[email_bool]

Character Classes

\[pdf\] → will select [pdf].

Character class	Pattern	Explanation
set	[fud]	Either f or u or d.
Range	[a-e]	Any of the characters a, b, c, d or e.
,,	[0-3]	Any of the characters 0, 1, 2 or 3.
,,	[A-Z]	Any Uppercase letter.
Set + Range	[A-Za-z]	Any Uppercase / lowercase letter.

Character Class	Pattern	Explanation
Digit	\d	Any digit character
Word	\w	Any digit, upper/lower case character or underscore.
White Space	\s	Any space, tab or line break.
Dot	.	Any character except newline.

Eg:- pattern = "\[\w+\]"

*A capture group can be created by enclosing a regular expression in parenthesis.

Eg:-

pattern = r"(\[\w+\])"

↳ raw-string.

* Negative character Classes:-

Character class	Pattern	Explanation
Negative set	[^fud]	Any character except f, u, d.
"	[^1-3Z\s]	Any character except 1, 2, 3, Z or white-space characters.
Negative Digit	\D	Any character except digit char.
Negative Word	\W	Any character except word char.
Negative whitespace	\S	Any character except whitespace characters.

Eg:- regex = r"[Jj]ava[^Ss]"

Word boundary anchor:- \b.

Eg:- ~~pattern = r"\J~~ pattern = r"\bJava\b"

flags:- re.I → Ignore Case.