RegEx

- A **regular expression** is a set of characters with highly specialized syntax that we can use to find or match other characters or groups of characters. In short, regular expressions, or Regex, are widely used in the UNIX world.

❖ Import the re Module

```
# Importing re module
import re
```

- The re-module in Python gives full support for regular expressions of Pearl style.
- The re module raises the re.error exception whenever an error occurs while implementing or using a regular expression.

❖ Metacharacters or Special Characters

Many letters have a particular meaning when utilized in a regular expression called metacharacters.

Characters	Meaning
.	**Dot** - It matches any characters except the newline character.
^	**Caret** - It is used to match the pattern from the start of the string. (Starts With)

$	**Dollar** - It matches the end of the string before the new line character. (Ends with)
*****	**Asterisk** - It matches zero or more occurrences of a pattern.
+	**Plus** - It is used when we want a pattern to match at least one.
?	**Question mark** - It matches zero or one occurrence of a pattern.
{}	**Curly Braces** - It matches the exactly specified number of occurrences of a pattern
[]	**Bracket** - It defines the set of characters
|	**Pipe** - It matches any of two defined patterns.

Example:

```
import re


s = 'Topper.World'


# without using \
match = re.search(r'.', s)
print(match)


# using \
match = re.search(r'\.', s)
print(match)
```

Output:

```
<re.Match object; span=(0, 1), match='T'>
<re.Match object; span=(6, 7), match='.'>
```

❖ Special Sequences:

- The ability to match different sets of symbols will be the first feature regular expressions can achieve that's not previously achievable with string techniques.
- On the other hand, Regexes isn't much of an improvement if that had been their only extra capacity. We can also define that some sections of the RE must be reiterated a specified number of times.
- The first metacharacter we'll examine for recurring occurrences is *. Instead of matching the actual character '*,' * signals that the preceding letter can be matched 0 or even more times rather than exactly once.
- Special Sequences consist of '\' followed by a character listed below. Each character has a different meaning.

Character	Meaning
\d	It matches any digit and is equivalent to [0-9].
\D	It matches any non-digit character and is equivalent to [^0-9].
\s	It matches any white space character and is equivalent to [\t\n\r\f\v]
\S	It matches any character except the white space character and is equivalent to [^\t\n\r\f\v]
\w	It matches any alphanumeric character and is equivalent to [a-zA-Z0-9]

\W	It matches any characters except the alphanumeric character and is equivalent to [^a-zA-Z0-9]
\A	It matches the defined pattern at the start of the string.
\b	r"\bxt" - It matches the pattern at the beginning of a word in a string. r"xt\b" - It matches the pattern at the end of a word in a string.
\B	This is the opposite of \b.
\Z	It returns a match object when the pattern is at the end of the string.

❖ Regex Module in Python

Python has a module named re that is used for regular expressions in Python. We can import this module by using the import statement.

- **compile** - It is used to turn a regular pattern into an object of a regular expression that may be used in a number of ways for matching patterns in a string.
- **search** - It is used to find the first occurrence of a regex pattern in a given string.
- **match** - It starts matching the pattern at the beginning of the string.
- **fullmatch** - It is used to match the whole string with a regex pattern.
- **split** - It is used to split the pattern based on the regex pattern.
- **findall** - It is used to find all non-overlapping patterns in a string. It returns a list of matched patterns.
- **finditer** - It returns an iterator that yields match objects.
- **sub** - It returns a string after substituting the first occurrence of the pattern by the replacement.
- **subn** - It works the same as 'sub'. It returns a tuple (new_string, num_of_substitution).
- **escape** - It is used to escape special characters in a pattern.
- **purge** - It is used to clear the regex expression cache.

❖ Match Object

A Match object contains all the information about the search and the result and if there is no match found then None will be returned. Let's see some of the commonly used methods and attributes of the match object.

➢ Getting the string and the regex

match.re attribute returns the regular expression passed and **match.string** attribute returns the string passed.

Example:

```
import re

s = "Welcome to Topper World"

# here x is the match object

res = re.search(r"\bG", s)
```

Output:

```
re.compile('\\bG')

Welcome to Topper World
```

➢ Getting index of matched object

- start() method returns the starting index of the matched substring
- end() method returns the ending index of the matched substring
- span() method returns a tuple containing the starting and the ending index of the matched substring