A RegEx, or Regular Expression, is a sequence of characters that forms a search pattern. RegEx can be used **to check if a string contains the specified search pattern**.

RegEx Functions

The `re` module offers a set of functions that allows us to search a string for a match:

Function	Description
findall	Returns a list containing all matches
search	Returns a Match object if there is a match anywhere in the string
split	Returns a list where the string has been split at each match
sub	Replaces one or many matches with a string

Metacharacters

Metacharacters are characters with a special meaning:

Character	Description	Example	Try it

[]	A set of characters	"[a-m]"	Try it »
\	Signals a special sequence (can also be used to escape special characters)	"\d"	Try it »
.	Any character (except newline character)	"he..o"	Try it »
^	Starts with	"^hello"	Try it »
$	Ends with	"planet$"	Try it »
*	Zero or more occurrences	"he.*o"	Try it »
+	One or more occurrences	"he.+o"	Try it »
?	Zero or one occurrences	"he.?o"	Try it »
{}	Exactly the specified number of occurrences	"he.{2}o"	Try it »

|	Either or	"falls|stays"	Try it »
()	Capture and group		

Special Sequences

A special sequence is a \ followed by one of the characters in the list below, and has a special meaning:

Character	Description	Example	Try it
\A	Returns a match if the specified characters are at the beginning of the string	"\AThe"	Try it »
\b	Returns a match where the specified characters are at the beginning or at the end of a word (the "r" in the beginning is making sure that the string is being treated as a "raw string")	r"\bain" r"ain\b"	Try it » Try it »
\B	Returns a match where the specified characters are present, but NOT at the beginning (or at the end) of a word	r"\Bain" r"ain\B"	Try it »

	(the "r" in the beginning is making sure that the string is being treated as a "raw string")		Try it »
\d	Returns a match where the string contains digits (numbers from 0-9)	"\d"	Try it »
\D	Returns a match where the string DOES NOT contain digits	"\D"	Try it »
\s	Returns a match where the string contains a white space character	"\s"	Try it »
\S	Returns a match where the string DOES NOT contain a white space character	"\S"	Try it »
\w	Returns a match where the string contains any word characters (characters from a to Z, digits from 0-9, and the underscore _ character)	"\w"	Try it »
\W	Returns a match where the string DOES NOT contain any word characters	"\W"	Try it »
\Z	Returns a match if the specified characters are at the end of the string	"Spain\Z"	Try it »

Sets

A set is a set of characters inside a pair of square brackets `[]` with a special meaning:

Set	Description	Try it
[arn]	Returns a match where one of the specified characters (`a`, `r`, or `n`) is present	Try it »
[a-n]	Returns a match for any lower case character, alphabetically between `a` and `n`	Try it »
[^arn]	Returns a match for any character EXCEPT `a`, `r`, and `n`	Try it »
[0123]	Returns a match where any of the specified digits (`0`, `1`, `2`, or `3`) are present	Try it »
[0-9]	Returns a match for any digit between `0` and `9`	Try it »
[0-5][0-9]	Returns a match for any two-digit numbers from `00` and `59`	Try it »
[a-zA-Z]	Returns a match for any character alphabetically between `a` and `z`, lower case OR upper case	Try it »

`[+]`	In sets, `+`, `*`, `.`, `|`, `()`, `$`,`{}` has no special meaning, so `[+]` means: return a match for any `+` character in the string

Where is regex used?

Regex, short for regular expression, is often used **in programming languages for matching patterns in strings, find and replace, input validation, and reformatting text**. Learning how to properly use Regex can make working with text much easier.

Why are we using regex?

Regular Expressions, also known as Regex, come in handy in a multitude of text processing scenarios. **Regex defines a search pattern using symbols and allows you to find matches within strings**. The applications of this span from software engineering to data science and beyond.

Which are 3 uses of regular expression?

Regular expressions are useful in any scenario that benefits from full or partial pattern matches on strings. These are some common use cases: **verify the structure of strings**. extract substrings from structured strings

Why do we need regex in Python?

Regular Expressions, also known as “regex” or “regexp”, are used **to match strings of text such as particular characters, words, or patterns of characters**. It means that we can match and extract any string pattern from the text with the help of regular expressions.

What are the components of regex?

Regular expression atoms

Single characters. A single character with no special significance represents that character in the target string. ...

Wild card. The . ...

Bracket Expressions. ...

Control characters. ...

Escape character sets. ...

Anchors. ...

Recursive expansion.

What is regex used for?

Short for regular expression, a regex is a string of text that **allows you to create patterns that help match, locate, and manage text**. Perl is a great example of a programming language that utilises regular expressions.

What are the benefits of regex?

Benefits of using Regular Expression

- Wide range of usage possibility, you may create one regular expression to validate any kind of input;
- Supported by almost any language, there are only a few programming languages which do not understand regular expressions;
- Do more with less, keep your code cleaner;

Can you show any real life scenario where regular expression helps?

Regular Expressions are useful for numerous practical day to day tasks that a data scientist encounters. They are used everywhere from **data pre-processing to natural language processing, pattern matching, web scraping, data extraction** and what not!

How to use RegEx for web scraping?
RegEx can be used **to validate all types of character combinations, including special characters like line breaks**.

What are regular expressions in Python?

A regular expression also known as regex is **a sequence of characters that defines a search pattern**. Regular expressions are used in search algorithms, search and replace dialogs of text editors, and in lexical analysis.. It is also used for input validation.