String Manipulation & Regular Expressions

Here is where my presentation begins

Table of contents

01

What is string manipulation?

02

Useful python string methods

03

Introduction to Regular Expressions (Regex)

04

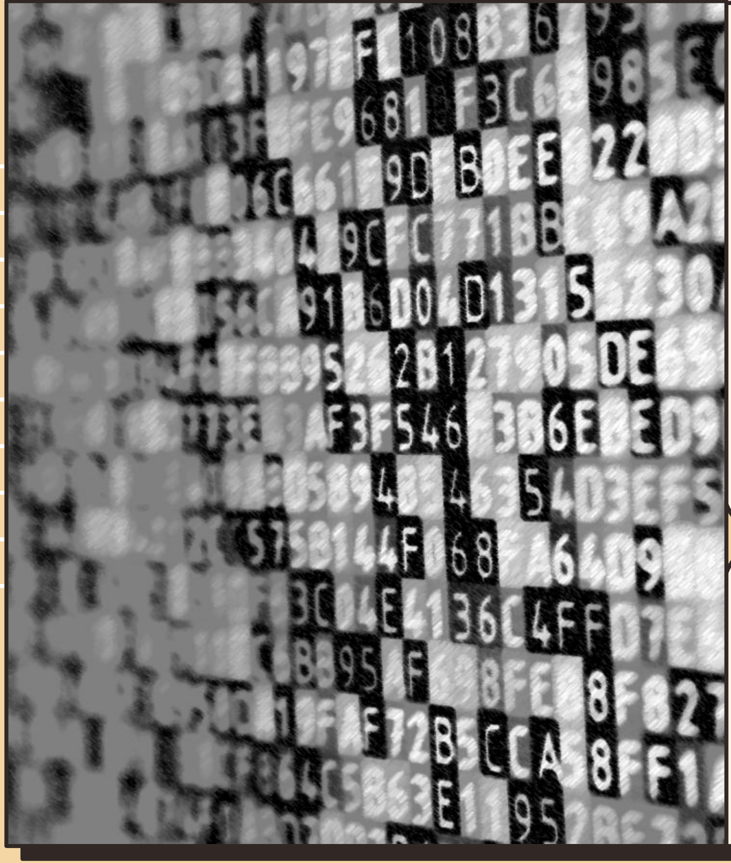
Regex syntax basics

05

Regex in python

06

Summary & tips



What is String Manipulation?

String manipulation is the process of working with text data (strings) to extract, modify, format, or analyze information. In programming, it involves operations like combining strings, changing their case, trimming spaces, or finding and replacing specific characters or words. It’s widely used in tasks like data cleaning, text processing, and user input handling.

Examples of string manipulation:

•Changing case: converting "hello" to "HELLO".
•Extracting substrings: getting "world" from "Hello World".
•Replacing characters: turning "Python3" into "Python".

Useful Python String Methods

Methods	Description	Example
.replace()	Replace substring	"abc".replace("a", "x") →“xbc”
.find()	Find index of substring	"hello".find(e) →1
.split()	split string into list	“abc”.split(“”) →[‘a’,’b’,’c’]
.join()	join list into string	“,”.join([‘a’,’b’]) →“a,b”

Introduction to Regular Expression(Regex)

Regular expressions (Regex) in Python are patterns used to match, search, and manipulate text efficiently. They help in tasks like validating inputs, extracting specific patterns, or replacing parts of a string. Python provides the “re” module to work with regular expressions.

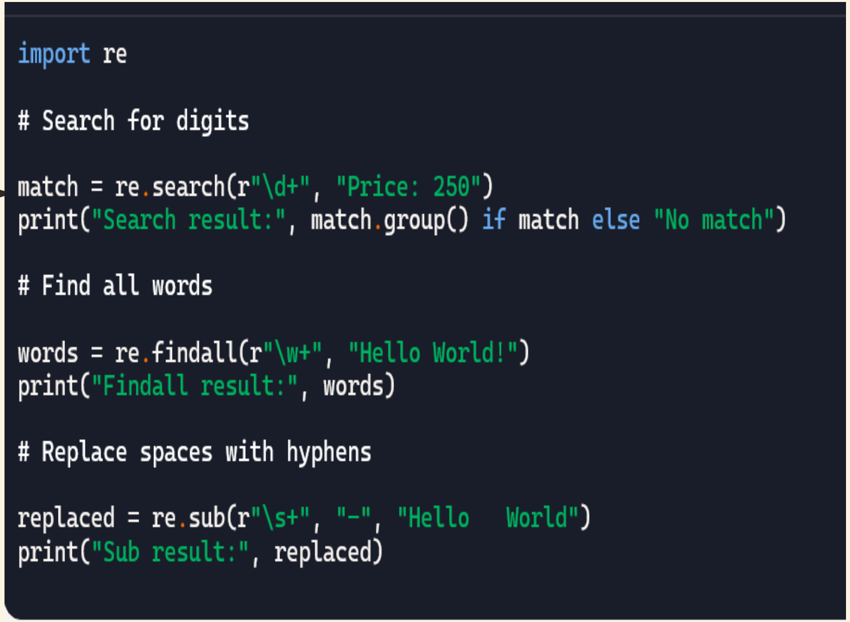
Examples of regex use:

* Validating formats, e.g., checking if an email or phone number is valid.
* Searching and extracting patterns, e.g., finding all numbers in a text.

Regex Syntax Basics

Symbol	Meaning	Example pattern
.	any character	a.b matches acb
^	Start of the string	^hello
$	End of the string	Bye$
\d	Digit	\d{3} matches 123
\w	Word character	\w+ matches hello
[]	Character set	[aeiou] matches vowel

Regex in python



```
import re

# Search for digits

match = re.search(r"\d+", "Price: 250")
print("Search result:", match.group() if match else "No match")

# Find all words

words = re.findall(r"\w+", "Hello World!")
print("Findall result:", words)

# Replace spaces with hyphens

replaced = re.sub(r"\s+", "-", "Hello   World")
print("Sub result:", replaced)
```

This Python script uses the re module (regular expressions) to perform three common text operations:

Search for digits:
re.search(r"\d+", "Price: 250") finds the first sequence of digits (250) in the string.

Find all words:
re.findall(r"\w+", "Hello World!") extracts all word-like elements (['Hello', 'World']).

Replace spaces with hyphens:
re.sub(r"\s+", "-", "Hello World") replaces one or more spaces with a single hyphen ('Hello-World').

Real-World Application

- **Form validation**: Email, phone number, password strength
- **Log parsing**: Extract timestamps, error codes
- **Web scraping**: Extract data from HTML
- **Data cleaning**: Remove unwanted characters or patterns

Summary & Tips

- Use string methods for quick edits
- Use regex for complex pattern matching
- Practice with real-world examples
- Try online tools like regex101.com