TOPS Technology

Python Fundamentals

Presented By:

Nandni Vala

String Manipulation

1.Understanding how to access and manipulate strings

- Strings in Python are immutable sequences of characters, which means you cannot modify them directly. However, you can access, slice, and create new strings using various operations and methods.
- **Accessing Strings**
- Strings can be accessed using indexing and slicing.
- **a. Indexing**
- Indexing allows access to individual characters of a string.
- Index starts at 0 for the first character and -1 for the last.
- s = "Hello, World!"
- print(s[0]) # First character: 'H'
- print(s[-1]) # Last character: '!'

- **Slicing**
- Slicing allows access to a substring by specifying a range.
- Syntax: string[start:end:condi]
- s = "Hello, World!"
- print(s[0:5]) # Substring from index 0 to 4: 'Hello'
- print(s[7:]) # Substring from index 7 to end: 'World!'
- print(s[::-1]) # Reverse the string: '!dlroW ,olleH'
- **String Concatenation and Repetition**
- **a. Concatenation**
- Combine two or more strings using the + operator.
- s1 = "Hello"
- s2 = "World"
- print(s1 + ", " + s2 + "!") # Output: 'Hello, World!'

- **Repetition**
- Repeat a string using the * operator.
- s = "Hi"
- print(s * 3) # Output: 'HiHiHi'
- **Common String Methods**
- **a. Case Conversion**

Method	Description	Example
s.lower()	Converts to lowercase	`"Hello".lower()` → `'hello'`
s.upper()	Converts to uppercase	`"hello".upper()` → `'HELLO'`
s.capitalize()	Capitalizes the first character	`"python".capitalize()` → `'Python'`
s.title()	Capitalizes first letter of each word	`"hello world".title()` → `'Hello World'`

➢b. Searching and Replacing:

Method	Description	Example
s.find(sub)	Finds first occurrence of `sub`	`"hello".find("l")` → `2`
s.replace(old, new)	Replaces `old` with `new`	`"hello".replace("l", "z")` → `'hezzo'`
s.startswith(prefix)	Checks if string starts with `prefix`	`"hello".startswith("he")` → `True`
s.endswith(suffix)	Checks if string ends with `suffix`	"hello".endwith("o")-->True

➢**Splitting and Joining:**

Method	Description	Example
s.split(separator)	Splits into list by `separator`	`"a,b,c".split(",")` → `['a', 'b', 'c']`
separator.join(iterable)	Joins iterable with `separator`	`",".join(['a', 'b', 'c'])` → `'a,b,c'`

2.Basic operations: concatenation, repetition, string methods (upper(), lower(), etc.).

- **Concatenation**
- Concatenation is the process of combining two or more strings using the + operator.
- **Example:**
- str1 = "Hello"
- str2 = "World"
- result = str1 + " " + str2 # Add a space between
- print(result) # Output: "Hello World"
- **Repetition**
- Repetition is achieved using the * operator. This repeats the string a specified number of times.
- **Example:**

- str1 = "Hi! "
- result = str1 * 3
- print(result) # Output: "Hi! Hi! Hi! "
- **String Methods for Case Conversion**
- Python strings come with built-in methods to change their case.
- **Examples:**

Method	Description	Example	Output
upper()	Converts all characters to uppercase	"hello".upper()	"HELLO"
lower()	Converts all characters to lowercase	"HELLO".lower()	"hello"
capitalize()	Capitalizes the first character	"python".capitalize()	"Python"
title()	Capitalizes the first letter of each word	"hello world".title()	"Hello World"
swapcase()	Swaps uppercase and lowercase characters	"Hello".swapcase()	"hELLO"

➢Additional String Methods

➢a. Finding and Replacing :

Method	Description	Example	Output
find(sub)	Finds the first occurrence of a substring	`"hello".find("l")`	2
replace(old, new)	Replaces all occurrences of `old` with `new`	"hello".replace("l", "z")	"hezzo"

➢b. Splitting and Joining :

Method	Description	Example	Output
split(separator)	Splits the string into a list using the separator	"a,b,c".split(",")	['a', 'b', 'c']
join(iterable)	Joins elements of an iterable into a string	",".join(['a', 'b', 'c'])	"a,b,c"

3.String slicing.

- String slicing is a way to extract a portion (substring) of a string by specifying a range of indices. It uses the syntax:
- string[start:end:step]
- **Parameters**
- **start**: (Optional) The starting index of the slice.
  - Default: 0 (the beginning of the string).
- **end**: (Optional) The ending index of the slice (exclusive).
  - Default: Length of the string (up to the end).
- **step**: (Optional) The step size to move between characters.
  - Default: 1.
  - **Examples :**
  - s = "Hello, World!"
  - print(s[0:12:2])