# String Formatting and Methods in Python

## ****String Formatting****

### 1. ****f-strings (formatted string literals)****

Introduced in Python 3.6, f-strings provide a concise and readable way to embed expressions inside string literals.

name = "Alice"

age = 30

print(f"My name is {name} and I am {age} years old.")

# Output: My name is Alice and I am 30 years old.

### 2. str.format() ****Method****

The format() method allows you to insert values into a string using placeholders.

name = "Alice"

age = 30

print("My name is {} and I am {} years old.".format(name, age))

# Positional

print("My name is {0} and I am {1} years old.".format(name, age))

# Named

print("My name is {name} and I am {age} years old.".format(name=name, age=age))

### 3. % ****Operator (Old-style Formatting)****

The % operator allows you to perform string formatting in an older style.

name = "Alice"

age = 30

print("My name is %s and I am %d years old." % (name, age))

### 4. ****Concatenation Using**** +

You can concatenate strings using the + operator.

name = "Alice"

greeting = "Hello, " + name + "!"

print(greeting)

# Output: Hello, Alice!

## ****String Methods****

### ****Case Manipulation****

* str.upper(): Converts all characters to uppercase.
* str.lower(): Converts all characters to lowercase.
* str.capitalize(): Capitalizes the first character of the string.
* str.title(): Capitalizes the first character of each word.
* str.swapcase(): Swaps the case of all characters.

**Example:**

s = "hello, world!"

print(s.upper()) # "HELLO, WORLD!"

print(s.lower()) # "hello, world!"

print(s.capitalize()) # "Hello, world!"

print(s.title()) # "Hello, World!"

print(s.swapcase()) # "HELLO, WORLD!"

### ****Search and Replace****

* str.find(sub[, start[, end]]): Returns the index of the first occurrence of sub.
* str.rfind(sub[, start[, end]]): Returns the index of the last occurrence of sub.
* str.index(sub[, start[, end]]): Like find(), but raises a ValueError if not found.
* str.rindex(sub[, start[, end]]): Like rfind(), but raises a ValueError if not found.
* str.replace(old, new[, count]): Replaces occurrences of old with new.

**Example:**

s = "hello, world!"

print(s.find("world")) # 7

print(s.rfind("l")) # 10

print(s.index("hello")) # 0

print(s.replace("world", "Python")) # "hello, Python!"

### ****Split and Join****

* str.split(sep=None, maxsplit=-1): Splits a string into a list using a separator.
* str.rsplit(sep=None, maxsplit=-1): Splits from the right.
* str.splitlines(keepends=False): Splits at line boundaries.
* str.join(iterable): Joins elements of an iterable with a separator.

**Example:**

s = "apple,banana,cherry"

print(s.split(",")) # ['apple', 'banana', 'cherry']

print(s.rsplit(",", 1)) # ['apple,banana', 'cherry']

print("Line1\nLine2".splitlines()) # ['Line1', 'Line2']

words = ["hello", "world"]

print(" ".join(words)) # "hello world"

### ****Whitespace Handling****

* str.strip([chars]): Removes leading and trailing characters.
* str.lstrip([chars]): Removes leading characters.
* str.rstrip([chars]): Removes trailing characters.

**Example:**

s = " hello "

print(s.strip()) # "hello"

print(s.lstrip()) # "hello "

print(s.rstrip()) # " hello"

### ****Character Inspection****

* str.isalpha(): Returns True if all characters are alphabetic.
* str.isdigit(): Returns True if all characters are digits.
* str.isalnum(): Returns True if all characters are alphanumeric.
* str.isspace(): Returns True if all characters are whitespace.
* str.islower(): Returns True if all characters are lowercase.
* str.isupper(): Returns True if all characters are uppercase.

**Example:**

s = "Python3"

print(s.isalpha()) # False (contains a digit)

print(s.isdigit()) # False (not all characters are digits)

print(s.isalnum()) # True (alphanumeric)

print(" ".isspace()) # True (only whitespace)

print("python".islower()) # True

print("PYTHON".isupper()) # True

### ****Encoding and Decoding****

* str.encode(encoding="utf-8", errors="strict"): Encodes a string into bytes.
* bytes.decode(encoding="utf-8", errors="strict"): Decodes bytes back into a string.

**Example:**

s = "hello"

encoded = s.encode("utf-8")

print(encoded) # b'hello'

decoded = encoded.decode("utf-8")

print(decoded) # "hello"