4 September

Strings in Python

Strings are sequences of characters enclosed within single quotes, double quotes, or triple quotes. They are one of the most commonly used data types in Python. Strings are immutable, which means once created, their contents cannot be changed. You can perform indexing, slicing, concatenation, and repetition on strings.

```
# Example of string creation
s1 = 'Hello'
s2 = "World"
s3 = '''This is a multi-line string.'''

print(s1, s2)
print(s3)
```

String Indexing and Slicing

Indexing allows accessing individual characters using their position (starting from 0). Negative indexing starts from the end (-1 for last character). Slicing is used to extract substrings using [start:end:step].

```
text = 'Python Programming'
print(text[0])      # First character
print(text[-1])     # Last character
print(text[0:6])    # 'Python'
print(text[7:])     # 'Programming'
print(text[::2])    # Every second character
```

Notes — continued

Common String Methods

Python provides many built-in string methods to perform operations like changing case, searching, replacing, etc. Some important methods include lower(), upper(), title(), strip(), replace(), split(), join(), and find().

```
s = '  Python Basics  '
print(s.lower())      # '  python basics  '
print(s.upper())      # '  PYTHON BASICS  '
print(s.strip())      # removes spaces from both ends
print(s.replace('Python', 'Java'))

words = 'apple,banana,grape'.split(',')
print(words)
joined = '-'.join(words)
print(joined)
```

String Formatting

String formatting is useful for displaying dynamic values inside strings. There are three ways: old-style using %, str.format(), and modern f-strings. f-strings are recommended because they are concise and readable.

```
name = 'Alice'
age = 20

# Using % operator
print('Name: %s, Age: %d' % (name, age))

# Using format()
print('Name: {}, Age: {}'.format(name, age))

# Using f-strings (best)
print(f'Name: {name}, Age: {age}')
```