Lecture 4: Text Data Type (`str`) in Python

1. Introduction to Strings (`str`)

A **string** in Python is a sequence of characters enclosed in quotes (either single `''`, double `""`, or triple quotes `'''`/`"""` for multi-line strings). Strings are one of the most commonly used data types in Python, and they allow us to work with text.

Example:

```
name = "Alice"  # A string enclosed in double quotes

greeting = 'Hello, World!'  # A string enclosed in single quotes
```

Strings are **immutable**, meaning that once a string is created, it cannot be changed.

2. Basic String Operations

Python provides a variety of operations that can be performed on strings, such as indexing, slicing, and concatenation. Let's explore them in detail.

---

3. String Indexing

String **indexing** allows us to access individual characters in a string using their position (or index). In Python, indexing starts at 0, meaning the first character of the string has an index of 0, the second character has an index of 1, and so on.

- **Positive Indexing**: Starts from the beginning (0, 1, 2, ...).
- **Negative Indexing**: Starts from the end (-1, -2, -3, ...).

Example:

```
text = "Python"

# Positive Indexing
print(text[0])  # Output: P (first character)
print(text[3])  # Output: h (fourth character)

# Negative Indexing
print(text[-1])  # Output: n (last character)
print(text[-3])  # Output: t (third character from the end)
```

In this example:

- `text[0]` refers to the first character 'P'.
- `text[-1]` refers to the last character 'n'.

---

4. String Slicing

Slicing allows us to extract a part (or substring) of a string by specifying a range of indices. The syntax for slicing is:

```
string[start:end]
```

Where `start` is the index where slicing begins, and `end` is the index where slicing stops (the character at the `end` index is **not included**).

Example:

```
text = "Python"

# Slicing from index 0 to 3 (not including 3)
print(text[0:3])  # Output: Pyt

# Slicing from index 1 to 4
print(text[1:4])  # Output: yth

# Slicing the whole string
print(text[:])  # Output: Python (entire string)

# Slicing with negative indices
print(text[-4:-1])  # Output: tho
```

In this example:

- `text[0:3]` extracts characters from index 0 to 2, resulting in "Pyt".
- `text[:3]` means the same as `text[0:3]`.

---

5. String Concatenation

Concatenation is the process of combining two or more strings into one. In Python, this is done using the `+` operator.

Example:

```
first_name = "John"
last_name = "Doe"

# Concatenating strings
full_name = first_name + " " + last_name
print(full_name)  # Output: John Doe
```

In this example, the two strings `first_name` and `last_name` are combined, with a space in between.

---

6. Other Common String Operations

Here are some other useful string operations in Python:

1. **String Length (`len()`)**: You can find the length (number of characters) of a string using the `len()` function.
   Example:

```
text = "Python"
print(len(text))  # Output: 6
```

2. **Changing Case: You can change the case of strings using the following methods:**

- **`.upper()`: Converts the string to uppercase.**
- **`.lower()`: Converts the string to lowercase.**
- **`.capitalize()`: Capitalizes the first letter of the string.**

Example:

```
text = "hello world"

print(text.upper())  # Output: HELLO WORLD
```

```
print(text.lower())  # Output: hello world

print(text.capitalize())  # Output: Hello world
```

3. Finding Substrings (`find()`): The `find()` method returns the index of the first occurrence of a substring. If the substring is not found, it returns -1.

Example:

```
text = "Hello, world!"

print(text.find("world"))  # Output: 7
```

4. Replacing Substrings (`replace()`): You can replace parts of a string using the `replace()` method.

Example:

```
text = "I love apples"

new_text = text.replace("apples", "oranges")

print(new_text)  # Output: I love oranges
```

7. Practice: Create String Manipulation Programs

Let's create some simple programs to practice string operations.

Program 1: String Indexing

```
# String indexing example
text = "Python"
print("First character:", text[0])  # Output: P
print("Last character:", text[-1])  # Output: n
```

Program 2: String Slicing

```
# String slicing example
text = "Programming"
print("First 5 characters:", text[:5])  # Output: Progr
print("Last 4 characters:", text[-4:])  # Output: ming
```

Program 3: String Concatenation

```
# String concatenation example
first_name = "Alice"
last_name = "Smith"
full_name = first_name + " " + last_name
print("Full name:", full_name)  # Output: Alice Smith
```

Program 4: String Case Conversion

```
# String case conversion example
text = "learning python is fun"
print("Uppercase:", text.upper())  # Output: LEARNING PYTHON IS FUN
print("Lowercase:", text.lower())  # Output: learning python is fun
print("Capitalized:", text.capitalize())  # Output: Learning python is fun
```

Program 5: Finding and Replacing in Strings

```
# Finding and replacing in a string
sentence = "I love programming"
index = sentence.find("love")
print("Index of 'love':", index)  # Output: 2

# Replacing 'programming' with 'Python'
new_sentence = sentence.replace("programming", "Python")
print(new_sentence)  # Output: I love Python
```

---

8. Summary

- **String indexing allows accessing individual characters using their positions.**
- **String slicing extracts parts of the string based on a range of indices.**
- **Concatenation combines two or more strings using the + operator.**
- **Strings offer various operations like converting case, finding substrings, and replacing parts of a string.**
- **Python strings are immutable, meaning once created, their content cannot be changed directly, but you can create new strings from them.**

By practicing these string operations, you'll gain a strong understanding of how to manipulate and work with text data in Python.

Practice :

Set 4: Text Data Type (`str`)

1. **Create a string variable and print its length.**
2. **Write a program to extract the first 5 characters of a string.**
3. **Reverse a string using slicing.**
4. **Concatenate two strings using +.**
5. **Convert a string to uppercase and lowercase.**
6. **Replace all spaces in a string with underscores.**
7. **Write a program that checks if a substring exists within a string.**
8. **Use a formatted string (f-string) to print a message including variables.**