Chapter 2: Advanced String Capabilities in Python

Tutorial by Ayes Chinmay

July 22, 2025

Contents

1 Introduction

This tutorial covers Chapter 2, ”Advanced String Capabilities,” focusing on Python’s powerful string manipulation features. You’ll learn about string immutability, conversions, operators, indexing, and various string methods. Each section includes explanations, code examples, and exercises.

2 Strings Are Immutable

Strings in Python are immutable, meaning their contents cannot be changed after creation. Any modification creates a new string.

```
text = "Hello"
text = text + " World"  # Creates a new string
print(text)
```

Output:

```
Hello World
```

Exercise 1: Create a string `greeting = "Hi"` and append ” there!” to it. Print the result.

3 Numeric Conversions, Including Binary

Convert strings to numbers using `int()`, `float()`, and convert numbers to binary strings using `bin()`.

```
num_str = "123"
num = int(num_str)
binary = bin(num)[2:]  # Remove '0b' prefix
print(f"Number: {num}, Binary: {binary}")
```

Output:

```
Number: 123, Binary: 1111011
```

Exercise 2: Convert the string ”255” to an integer and print its binary representation.

4 String Operators (+, =, *, >, etc.)

String operators include concatenation (+), repetition (*), and comparison (>, <, ==).

```
str1 = "Hello"
str2 = "World"
concat = str1 + " " + str2
repeat = str1 * 2
print(f"Concatenation: {concat}")
print(f"Repetition: {repeat}")
print(f"Is str1 > str2? {str1 > str2}")
```

Output:

```
Concatenation: Hello World
Repetition: HelloHello
Is str1 > str2? False
```

Exercise 3: Concatenate two strings and repeat the result three times.

5 Indexing and Slicing

Strings are indexed (0-based) and can be sliced using `[start:end:step]`.

```
text = "Python"
print(f"First char: {text[0]}")
print(f"Slice [1:4]: {text[1:4]})
print(f"Reverse: {text[::-1]}")
```

Output:

```
First char: P
Slice [1:4]: yth
Reverse: nohtyP
```

Exercise 4: Extract the substring "gram" from "Programming".

6 Single-Character Functions (Character Codes)

Use `ord()` to get a character's Unicode code point and `chr()` to convert a code point to a character.

```
char = "A"
code = ord(char)
back = chr(code)
print(f"Character {char} has code {code}")
print(f"Code {code} is character {back}")
```

Output:

```
Character A has code 65
Code 65 is character A
```

Exercise 5: Find the Unicode code point of 'z' and convert 97 back to a character.

7 Building Strings Using "join"

The `join()` method concatenates a list of strings with a separator.

```
words = ["Hello", "World"]
sentence = " ".join(words)
print(sentence)
```

Output:

```
Hello World
```

Exercise 6: Join the list `["apple", "banana", "orange"]` with a comma and space.

8 Important String Functions

Key string methods include `len()`, `upper()`, `lower()`, and `strip()`.

```
text = "  Python  "
print(f"Length: {len(text)}")
print(f"Upper: {text.upper()}")
```

```python
print(f"Lower: {text.lower()}")
print(f"Stripped: '{text.strip()}'")
```

Output:

```
Length: 9
Upper:   PYTHON
Lower:   python
Stripped: 'Python'
```

Exercise 7: Apply `upper()` and `strip()` to " hello " and print the result.

9 Binary, Hex, and Octal Conversion Functions

Convert numbers to binary (`bin()`), hexadecimal (`hex()`), or octal (`oct()`).

```python
num = 42
print(f"Binary: {bin(num)[2:]}")
print(f"Hex: {hex(num)[2:]}")
print(f"Octal: {oct(num)[2:]}")
```

Output:

```
Binary: 101010
Hex: 2a
Octal: 52
```

Exercise 8: Convert 100 to hexadecimal and octal.

10 Simple Boolean ("is") Methods

Methods like `isalpha()`, `isdigit()`, and `isspace()` test string properties.

```python
text = "Python123"
print(f"Is alpha? {text.isalpha()}")
print(f"Is digit? {text.isdigit()}")
print(f"Is alphanumeric? {text.isalnum()}")
```

Output:

```
Is alpha? False
Is digit? False
Is alphanumeric? True
```

Exercise 9: Check if "123" is a digit and if "abc" is alphabetic.

11 Case Conversion Methods

Methods like `upper()`, `lower()`, `title()`, and `capitalize()` modify string case.

```python
text = "hello world"
print(f"Title: {text.title()}")
print(f"Capitalize: {text.capitalize()}")
```

Output:

```
Title: Hello World
Capitalize: Hello world
```

Exercise 10: Convert ”python programming” to title case.

12 Search-and-Replace Methods

Use `find()`, `replace()`, and `count()` for searching and replacing.

```python
text = "Hello World"
print(f"Find 'World': {text.find('World')}")
print(f"Replace: {text.replace('World', 'Python')}")
print(f"Count 'l': {text.count('l')}")
```

Output:

```
Find 'World': 6
Replace: Hello Python
Count 'l': 3
```

Exercise 11: Replace ”cat” with ”dog” in ”The cat is on the mat”.

13 Breaking Up Input Using “split”

The `split()` method splits a string into a list based on a delimiter.

```python
text = "apple,banana,orange"
fruits = text.split(",")
print(fruits)
```

Output:

```
['apple', 'banana', 'orange']
```

Exercise 12: Split ”one two three” on spaces.

14 Stripping

Methods like `strip()`, `lstrip()`, and `rstrip()` remove whitespace.

```python
text = "  Hello  "
print(f"Strip: '{text.strip()}'")
print(f"Left strip: '{text.lstrip()}'")
print(f"Right strip: '{text.rstrip()}'")
```

Output:

```
Strip: 'Hello'
Left strip: 'Hello  '
Right strip: '  Hello'
```

Exercise 13: Strip whitespace from ” Python ”.

15 Justification Methods

Methods like `center()`, `ljust()`, and `rjust()` align text.

```
text = ”Python”
print(f”Center: '{text.center(10, '*')}'”)
print(f”Left justify: '{text.ljust(10)}'”)
print(f”Right justify: '{text.rjust(10)}'”)
```

Output:

```
Center: '**Python**'
Left justify: 'Python    '
Right justify: '    Python'
```

Exercise 14: Center ”Code” in a 10-character field with dashes.

16 Summary

This chapter covered advanced string manipulation in Python, including immutability, conversions, operators, indexing, and various methods for processing strings. Practice the exercises to master these concepts.

17 Answers to Exercises

1. **Exercise 1**:

```
greeting = ”Hi”
greeting = greeting + ” there!”
print(greeting)
```

2. **Exercise 2**:

```
num_str = ”255”
num = int(num_str)
print(bin(num)[2:])
```

3. **Exercise 3**:

```
str1 = ”Hello”
str2 = ”World”
result = (str1 + ” ” + str2) * 3
print(result)
```

4. **Exercise 4**:

```
text = ”Programming”
print(text[3:7])
```

5. **Exercise 5**:

```
print(ord('z'))
print(chr(97))
```

6. **Exercise 6**:

```
fruits = ["apple", "banana", "orange"]
print(", ".join(fruits))
```

7. **Exercise 7**:

```
text = "  hello  "
print(text.strip().upper())
```

8. **Exercise 8**:

```
num = 100
print(hex(num)[2:])
print(oct(num)[2:])
```

9. **Exercise 9**:

```
print("123".isdigit())
print("abc".isalpha())
```

10. **Exercise 10**:

```
text = "python programming"
print(text.title())
```

11. **Exercise 11**:

```
text = "The cat is on the mat"
print(text.replace("cat", "dog"))
```

12. **Exercise 12**:

```
text = "one two three"
print(text.split())
```

13. **Exercise 13**:

```
text = "   Python   "
print(text.strip())
```

14. **Exercise 14**:

```
text = "Code"
print(text.center(10, '-'))
```