Python Strings

Quick Reference Guide

Essential Methods, Patterns & Best Practices

For CTE Computer Science Students

📚 Table of Contents

- String Basics
- Common Methods
- String Slicing
- Checking & Testing
- Searching & Finding
- String Formatting
- Common Patterns
- Best Practices

String Basics

Key Concept: Strings are immutable sequences of characters.

```python
name = "Bergen Tech"
length = len(name)  # 11

# Accessing characters
first_char = name[0]    # "B"
last_char = name[-1]    # "h"

# Strings are immutable
name[0] = "b"  # ❌ ERROR!
name = "bergen tech"  # ✅ CORRECT
```

Case Conversion

`.upper()` - Convert to UPPERCASE

```python
text = "Hello"
text.upper()  # "HELLO"
```

`.lower()` - Convert to lowercase

```python
text = "Hello"
text.lower()  # "hello"
```

Use .lower() for case-insensitive comparisons!

More Case Methods

`.title()` - Title Case Each Word

```
text = "hello world"
text.title()  # "Hello World"
```

`.capitalize()` - Capitalize first letter only

```
text = "hello world"
text.capitalize()  # "Hello world"
```

Whitespace Methods

`.strip()` - Remove leading & trailing whitespace

```
text = "  Hello World  "
text.strip()   # "Hello World"
```

`.lstrip()` - Remove LEFT whitespace

```
text = "  Hello  "
text.lstrip()  # "Hello  "
```

`.rstrip()` - Remove RIGHT whitespace

```
text = "  Hello  "
text.rstrip()  # "  Hello"
```

Searching Methods

`in` **keyword** - Check if substring exists

```
text = "Bergen Tech CTE"
"Tech" in text      # True
"tech" in text      # False (case-sensitive!)
```

`.find(substring)` - Returns index or -1

```
text = "Hello World"
text.find("World")  # 6
text.find("xyz")    # -1
```

More Searching Methods

`.index(substring)` - Returns index or ERROR

```
text = "Hello World"
text.index("World") # 6
text.index("xyz")   # ERROR!
```

Prefer "in" for checking, .find() for position

Checking Methods (True/False)

`.startswith()` & `.endswith()`

```
email = "test@gmail.com"
email.startswith("test")  # True

file = "document.pdf"
file.endswith(".pdf")     # True
```

`.isdigit()` & `.isalpha()`

```
"123".isdigit()     # True
"Hello".isalpha()   # True
"Hello123".isalpha()  # False
```

More Checking Methods

`.isalnum()` - Letters or digits only

```python
"Hello123".isalnum()  # True
"Hello 123".isalnum() # False (space!)
```

`.isspace()` - Only whitespace

```python
"   ".isspace()  # True
"".isspace()     # False
```

Replacing & Counting

`.replace(old, new)` - Replace ALL occurrences

```
text = "Hello World Hello"
text.replace("Hello", "Hi")  # "Hi World Hi"
```

`.count(substring)` - Count occurrences

```
text = "banana"
text.count("a")    # 3
text.count("na")   # 2
```

Splitting & Joining

`.split(separator)` - Split string into list

```
text = "Hello World Python"
words = text.split()  # ["Hello", "World", "Python"]
```

`.join(list)` - Join list into string

```
words = ["Hello", "World"]
" ".join(words)   # "Hello World"
"-".join(words)   # "Hello-World"
```

String Slicing Basics

Syntax: `string[start:end:step]`

```
text = "Python Programming"

text[0:6]       # "Python" (index 0 to 5)
text[7:18]      # "Programming"

# Omitting start/end
text[:6]        # "Python"
text[7:]        # "Programming"
```

Advanced Slicing

```
text = "Python Programming"

# Negative indices
text[-11:]     # "Programming" (last 11 chars)

# Step (every nth character)
text[::2]      # "Pto rgamn" (every 2nd char)

# Reverse string
text[::-1]     # "gnimmargorP nohtyP"
```

String Formatting with F-Strings

F-strings are the modern Python way! (Python 3.6+)

```python
name = "Alice"
age = 20
message = f"Hello, {name}! You are {age} years old."

# Expressions inside f-strings
x, y = 10, 5
result = f"{x} + {y} = {x + y}"  # "10 + 5 = 15"

# Method calls
text = "python"
formatted = f"Language: {text.upper()}"  # "Language: PYTHON"
```

Pattern: Character-by-Character

Use when: Filtering or transforming each character

```python
def extract_digits(text):
    result = ""
    for char in text:
        if char.isdigit():
            result += char
    return result

phone = "Call: 201-555-1234"
digits = extract_digits(phone)  # "2015551234"
```

Pattern: Split-Process-Join

Use when: Processing words individually

```python
def title_case_custom(text):
    words = text.split()           # Split
    capitalized = []               # Process

    for word in words:
        cap_word = word[0].upper() + word[1:].lower()
        capitalized.append(cap_word)

    return " ".join(capitalized)   # Join

title_case_custom("hello WORLD")  # "Hello World"
```

Pattern: Method Chaining

Use when: Multiple operations in sequence

```python
def clean_user_input(text):
    return text.strip().lower()

username = "  JohnDoe123  "
clean = clean_user_input(username)  # "johndoe123"

def create_url_slug(title):
    return title.strip().lower().replace(" ", "-")

create_url_slug("  My Blog Post  ")  # "my-blog-post"
```

🌟 Best Practices

1. Always normalize user input early

```python
def process_search(query):
    query = query.strip().lower()  # Do this FIRST!
```

2. Use f-strings for formatting

```python
# ✅ Modern and readable
name = "Alice"
message = f"Hello, {name}!"
```

⚠️ Common Mistakes

Mistake #1: Forgetting strings are immutable

```python
# ❌ WRONG
name = "john"
name.upper()  # Returns "JOHN" but name unchanged

# ✅ CORRECT
name = name.upper()  # Reassign
```

Mistake #2: Case-sensitive comparisons

```python
# ❌ WRONG
email = "User@Gmail.COM"
if email == "user@gmail.com":  # False!

# ✅ CORRECT
if email.lower() == "user@gmail.com":  # True!
```

📋 Quick Reference

Method	Purpose	Example
`.upper()`	UPPERCASE	`"hi".upper()` → `"HI"`
`.lower()`	lowercase	`"HI".lower()` → `"hi"`
`.strip()`	Remove whitespace	`"  hi  ".strip()` → `"hi"`
`.replace()`	Replace text	`"hi".replace("h", "H")` → `"Hi"`
`.split()`	Split to list	`"a b".split()` → `["a", "b"]`

💪 Practice Problems

1. **Username Validator**: 5-15 chars, alphanumeric only
2. **Email Masker**: Convert `john@email.com` to `j***@email.com`
3. **Palindrome Checker**: Reads same forwards/backwards
4. **Word Counter**: Count words in a sentence
5. **Initials Generator**: Extract first letter of each word

🎯 Key Takeaways

Strings are immutable - methods return NEW strings!

Essential Methods:

- `.upper()`, `.lower()`, `.strip()` - Cleaning
- `.split()`, `.join()` - String/list conversion
- `.replace()`, `.find()`, `in` - Search/modify
- `[start:end]` - Slicing substrings