# Strings Cheat Sheet

Reference: https://docs.python.org/3/library/stdtypes.html#string-methods

Prepared By: Shehab Abdel-Salam

## Common String Methods in Python

1. `len()`: Returns the length of the string.

```
s = "Hello"
print(len(s))  # Output: 5
```

2. `str.lower()`: Converts all characters in the string to lowercase.

```
s = "Hello"
print(s.lower())  # Output: "hello"
```

3. `str.upper()`: Converts all characters in the string to uppercase.

```
s = "Hello"
print(s.upper())  # Output: "HELLO"
```

4. `str.title()`: Converts the first character of each word to uppercase.

```
s = "hello world"
print(s.title())  # Output: "Hello World"
```

5. `str.strip()`: Removes leading and trailing whitespaces.

```
s = "  hello  "
print(s.strip())  # Output: "hello"
```

6. `str.replace(old, new)`: Replaces occurrences of `old` substring with `new`.

```
s = "Hello world"
print(s.replace("world", "Python"))  # Output: "Hello Py
thon"
```

7. `str.split(separator)`: Splits the string into a list of substrings based on the separator.

```
s = "Hello,world"
print(s.split(","))  # Output: ["Hello", "world"]
```

8. `str.partition(separator)`: Splits the string into a 3-tuple containing the part before the separator, the separator itself, and the part after the separator.

```
s = "PythonCheatSheet"
print(s.partition("Cheat")) # Output: ('Python', 'Chea
t', 'Sheet')
```

9. `str.join(iterable)`: Joins elements of an iterable into a single string, separated by the string.

```
lst = ["Hello", "world"]
print(" ".join(lst))  # Output: "Hello world"
```

10. `str.find(sub)`: Returns the index of the first occurrence of the substring `sub`. Returns -1 if not found.

```
s = "Hello world"
print(s.find("world"))  # Output: 6
```

11. `str.index(sub)`: Same as `str.find(sub)`.
Returns
`ValueError` exception if not found.

```
s = "Hello world"
print(s.find("world"))  # Output: 6
```

12. `str.startswith(prefix)`: Checks if the string starts with the specified prefix.

```
s = "Hello world"
print(s.startswith("Hello"))  # Output: True
```

13. `str.endswith(suffix)`: Checks if the string ends with the specified suffix.

```
s = "Hello world"
print(s.endswith("world"))  # Output: True
```

14. `str.removeprefix(prefix)`: If the string starts with the prefix string, return `string[len(prefix):]`. Otherwise, return a copy of the original string.

```
s = "HelloWorld"
print(s.removeprefix("Hello"))  # Output: World
print(s.removeprefix("World"))  # Output: HelloWorld
```

15. `str.removesuffix(suffix)`: If the string ends with the suffix string and is is not empty, return `string[:-len(prefix)]`. Otherwise, return a copy of the original string.

```
s = "HelloWorld"
print(s.removesuffix("World"))  # Output: Hellow
print(s.removesuffix("Hellow"))  # Output: HelloWorld
```

12. **(Not recommended) Using `str.format()` for formatted strings**:

```
name = "Alice"
age = 30
s = "Name: {}, Age: {}".format(name, age)
print(s)  # Output: "Name: Alice, Age: 30"
```

13. **(Recommended) Using f-strings (formatted string literals)**:

```
name = "Alice"
age = 30
s = f"Name: {name}, Age: {age}"
print(s)  # Output: "Name: Alice, Age: 30"
```

14. **Multi-line strings using triple quotes**:

```
s = """This is a
multi-line
string."""
print(s)
# Output:
# This is a
# multi-line
# string.
```

15. **String slicing**:

```
s = "Hello world"
print(s[0:5])  # Output: "Hello"
print(s[6:])   # Output: "world"
print(s[-5:])  # Output: "world"
```

16. **String concatenation**:

```
s1 = "Hello"
s2 = "world"
s = s1 + " " + s2
print(s)  # Output: "Hello world"
```

17. **Escaping characters**:

```
s = "He said, \"Hello, world!\""
print(s)  # Output: 'He said, "Hello, world!"'
```

18. **Raw strings**:

```
s = r"C:\Users\name"
print(s)  # Output: "C:\Users\name"
```

19. **Changing case using `str.capitalize()` and `str.swapcase()`**:

```
s = "hello world"
print(s.capitalize())  # Output: "Hello world"
```

```
print(s.swapcase())    # Output: "HELLO WORLD"
```

20. **Checking content with `str.isdigit()`, `str.isalpha()`, `str.isalnum()`, `str.isnumeric()`:**

```
# Checks for digit characters (0-9) only.
s = "12345"
print(s.isdigit())  # Output: True

# Checks if all chars are alphabetic and there is at lea
st one character.
s = "hello"
print(s.isalpha())  # Output: True

# Checks if all chars are alphanumeric and there is at l
east one character.
s = "hello123"
print(s.isalnum())  # Output: True

# Checks for any numeric characters, including non-decim
al numbers.
s= "½"
print(s.isnumeric())       # Output: True
```

21. **Padding strings with `str.zfill(width)` and `str.rjust(width)`:**

```
s = "42"
print(s.zfill(5))  # Output: "00042"

s = "42"
print(s.rjust(5, '0'))  # Output: "00042"
```