Python Strings - MAD 102 Week 7 Notes

Hia Al Saleh

October 17th, 2024

Contents

1 Introduction to Strings in Python **2**
- 1.1 Indexing and Slicing . . . 2
- 1.2 String Length . . . 2

2 Slicing Strings **2**

3 String Formatting **2**
- 3.1 Field Width and Alignment . . . 3
- 3.2 Fill Characters . . . 3

4 String Methods **3**
- 4.1 replace() . . . 3
- 4.2 find() . . . 3

5 Comparing Strings **3**
- 5.1 Membership and Identity Operators . . . 4

6 Looping Through Strings **4**

7 String Validation Methods **4**

8 String Manipulation Methods **4**

9 String Splitting and Joining **5**

10 Password Validation Challenge **5**

1 Introduction to Strings in Python

Strings are one of the most commonly used data types in Python. They are sequences of characters enclosed in either single or double quotes.

```
# Initialize a string
string = "Hello World!"
print(string[0:4])  # Outputs 'Hell'
```

We can use indexing and slicing to extract parts of a string. Python supports both positive and negative indexing.

1.1 Indexing and Slicing

Indexing starts from 0, and negative indexing begins from -1 (last character).

```
# Positive indexing
string[1]  # Outputs 'e'

# Negative indexing
string[-1]  # Outputs '!'
```

1.2 String Length

The length of a string can be determined using the `len()` function:

```
len(string)  # Outputs 12
```

2 Slicing Strings

Slicing allows us to extract a range of characters from a string:

```
word = 'batman'
print(word[3:5])  # Outputs 'ma'
print(word[-3:])  # Outputs 'man'
```

The start or end of the slice can be omitted:

```
print(word[3:])   # Outputs 'man'
print(word[:3])   # Outputs 'bat'
```

3 String Formatting

Python provides several methods for formatting strings:

3.1 Field Width and Alignment

We can use formatted string literals (f-strings) to specify field width and alignment:

```
print(f'{"Student Name":20}{"Marks":^10}')
# Output:
# Student Name          Marks
# ----------------------------
# Malcola Reynolds        60
```

3.2 Fill Characters

Padding characters can be specified in the format:

```
print(f'{10:0>6}')  # Outputs '000010'
```

4 String Methods

Python strings support a variety of useful methods, including:

4.1 replace()

The `replace()` method replaces parts of a string with another substring:

```
title = 'The new adventures of Indiana Jones'
print(title.replace('new', 'continuing'))
# Outputs: 'The continuing adventures of Indiana Jones'
```

4.2 find()

The `find()` method returns the index of the first occurrence of a substring:

```
phrase = 'This is very, very, very long'
print(phrase.find('very'))  # Outputs: 8
```

5 Comparing Strings

Python supports comparison of strings using relational and equality operators:

```
print('a' > 'A')  # Outputs: True
print('bat' > 'ball')  # Outputs: True
```

5.1 Membership and Identity Operators

We can check for substring membership or identity:

```
print('bat' in 'batman')  # Outputs: True
print(string1 is string2)  # Checks if both variables
    refer to the same object
```

6 Looping Through Strings

We can iterate through strings using a `for` loop:

```
word = 'batman'
for char in word:
    print(char)
# Outputs:
# b
# a
# t
# m
# a
# n
```

7 String Validation Methods

Python provides several methods for validating string content:

```
print('abc123'.isalnum())  # Returns True if all
    characters are alphanumeric
print('123'.isdigit())  # Returns True if all characters
    are digits
print('abc'.islower())  # Returns True if all characters
    are lowercase
print('   '.isspace())  # Returns True if all characters
    are whitespace
```

8 String Manipulation Methods

Common string manipulation methods include:

- `capitalize()` – Capitalizes the first character.
- `lower()` – Converts all characters to lowercase.
- `upper()` – Converts all characters to uppercase.
- `strip()` – Removes leading and trailing spaces.

- `title()` – Capitalizes the first letter of each word.

```
phrase = "  frozen is my FAVOURITE movie!!!    "
print(phrase.capitalize())  # Outputs: 'Frozen is my
    favourite movie!!!'
print(phrase.strip())  # Outputs: 'frozen is my FAVOURITE
    movie!!!'
```

9 String Splitting and Joining

The `split()` method splits a string into a list of substrings based on a separator:

```
phrase = "I love to watch Frozen, Despicable Me, Free
    Birds"
print(phrase.split(','))  # Outputs: ['I love to watch
    Frozen', ' Despicable Me', ' Free Birds ']
```

The `join()` method joins a list of strings with a specified separator:

```
list = ['https', 'www', 'google', 'com']
print('/'.join(list))  # Outputs: 'https/www/google/com'
```

10 Password Validation Challenge

A sample challenge involves validating a password based on certain criteria:

```
email = input("Enter Email ID: ").lower().strip()
password = input("Enter Password: ")

if (len(password) >= 8 and
    any(char.islower() for char in password) and
    any(char.isupper() for char in password) and
    any(char.isdigit() for char in password)):
    print("Password is valid")
else:
    print("Password is invalid")
```

This program ensures the password contains at least 8 characters, one lowercase, one uppercase, and one digit.