Class 02: Strings in Python

- Python Strings
- Slicing Strings
- Modify Strings
- Concatenate Strings
- Format Strings

Strings

Strings in python are surrounded by either single quotation marks, or double quotation marks.

'hello' is the same as "hello".

You can display a string literal with the `print()` function:

Example

```
# Using single quotes
single_quote_string = 'hello'
print(single_quote_string)

# Using double quotes
double_quote_string = "hello"
print(double_quote_string)
```

Strings are Arrays

Like many other popular programming languages, strings in Python are arrays of bytes representing unicode characters.

However, Python does not have a character data type, a single character is simply a string with a length of 1.

Square brackets can be used to access elements of the string

```
a = "Hello, World!"

print(a[1])
```

String Length

To get the length of a string, use the `len()` function.

Example

The `len()` function returns the length of a string:

```
a = "Hello, World!"

print(len(a))
```

Slicing

You can return a range of characters by using the slice syntax.

Specify the start index and the end index, separated by a colon, to return a part of the string.

```
b = "Hello, World!"
print(b[2:5])
```

Slice From the Start

By leaving out the start index, the range will start at the first character:

Example

```
b = "Hello, World!"
print(b[:5])
```

Slice To the End

By leaving out the *end* index, the range will go to the end:

Example

```
b = "Hello, World!"
print(b[2:])
```

Negative Indexing

Use negative indexes to start the slice from the end of the string:

```
b = "Hello, World!"
print(b[-5:-2])
```

Python - Modify Strings

Python has a set of built-in methods that you can use on strings.

```
a = "sayem"
print(a.upper())
```

```
a = "SAYEM"
print(a.lower())
```

Remove Whitespace

Whitespace is the space before and/or after the actual text, and very often you want to remove this space.

Example

The `strip()` method removes any whitespace from the beginning or the end:

```
a = " Hello, World! "
print(a.strip()) # returns "Hello, World!"
```

Python - String Concatenation

To concatenate, or combine, two strings you can use the + operator.

Example

```
a = "Hello"
b = "World"
c = a + b
print(c)
```

Example

To add a space between them, add a " ":

```
a = "Hello"
b = "World"
c = a + " " + b
print(c)
```

Python - Format - Strings

String Format

As we learned in the Python Variables chapter, we cannot combine strings and numbers like this:

```
name = "sayem"
age = 27
```

```
print(f"my name is{name} and my age is {age}")
```