String

Monday, December 13, 2021 2:53 PM

STRING DATA TYPE

A String is an array of characters. String variables can be declared using either single or double quotation marks. More advanced strings can use triple single or double quotes (see string interpellation section). Note, Python does not require a semicolon at the end unlike languages such as Java or C#.

The size of a string is limited to system memory, see example below:

```
In [3]: string1 = "Hello, I am teaching you Python"
        print(string1)

Out[4]: Hello, I am teaching you Python
```

The following is the same example but with the string casted:

```
In [9]: message = str("Hello, I am teaching you Python")
        print(message)

Out[4]: Hello, I am teaching you Python
```

Additionally, using \n indicates the string that follows will be on a new line.

DOCUMENTATION STRING

Strings can also use triple quotes, this is called a documentation string or a "doc string" and is placed after the first line of a class or defined function and is usually used as help text. It can be called using the `help()` function for structures such as classes which will be explored later on, see this basic example of a doc string:

```
class User:
    """This is a doc string"""
```

You can see the value stored in the variable by printing it. In this case the variable name gets put in the parenthesis of the `print()` method.

MULTI QUOTE STRINGS

The reasoning for having two ways to write strings using either single or double quotation is marks is to support words that use apostrophes, see example.

```
In [11]: message = str('I'm good how are you?')

Out[10]:   File "C:\Users\Joe\AppData\Local\Temp/ipykernel_7984/3229079199.py", line 1
             message = str('I'm good how are you?')
                           ^
           SyntaxError: invalid syntax
```

This code block raised an error because the Python interpreter thinks the apostrophe in the word I'm is the end of the string, thus the remaining text causes a syntax error.

A method of getting around this is to use a backslash `\` Infront of the apostrophe in the word I'm `I\'m`. This tells the Python interpreter that the quote is treated as a character and not the end of the string.

The better alternative way is to use double quotes instead, see the following example.

```
In [14]: message = str("I'm good how are you?")
         print(message)

Out[15]: "I'm good how are you?"
```

If you create a string with double quotes that contains text that have a quotation mark, you will encounter another error because the Python interpreter thinks the double quote before the world I is the end of the string. Sound familiar?. See the below example.

```
In [16]: message = str("He told me "I love ice cream" yesterday")
         print(message)

Out[15]:   File "C:\Users\Joe\AppData\Local\Temp/ipykernel_7984/2514675575.py", line 1
             message = str("He told me "I love ice cream" yesterday")
                                        ^
         SyntaxError: invalid syntax
```

This syntax error can be avoided by using single quotes instead, see example.

```
In [17]: message = str('He told me "I love ice cream" yesterday')
         print(message)

Out[18]: 'He told me "I love ice cream" yesterday'
```