**Python Strings**

In Python, a string is a sequence of characters. For example, "hello" is a string containing a sequence of characters 'h', 'e', 'l', 'l', and 'o'.

We use single quotes or double quotes to represent a string in Python. For example,

# create a string using double quotes

string1 = "Python programming"

# create a string using single quotes

string1 = 'Python programming'

Here, we have created a string [variable](https://www.programiz.com/python-programming/variables-constants-literals) named string1. The variable is initialized with the string "Python Programming".

**Example: Python String**

# create string type variables

name = "Python"

print(name)

message = "I love Python."

print(message)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

Python

I love Python.

In the above example, we have created string-type variables: name and message with values "Python" and "I love Python" respectively.

Here, we have used double quotes to represent strings, but we can use single quotes too.

**Access String Characters in Python**

We can access the characters in a string in three ways.

* **Indexing:** One way is to treat strings as a [list](https://www.programiz.com/python-programming/list) and use index values. For example,

greet = 'hello'

# access 1st index element

print(greet[1]) # "e"

[Run Code](https://www.programiz.com/python-programming/online-compiler)

* **Negative Indexing**: Similar to a list, Python allows negative indexing for its strings. For example,

greet = 'hello'

# access 4th last element

print(greet[-4]) # "e"

[Run Code](https://www.programiz.com/python-programming/online-compiler)

* **Slicing:** Access a range of characters in a string by using the slicing operator colon :. For example,

greet = 'Hello'

# access character from 1st index to 3rd index

print(greet[1:4]) # "ell"

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Note**: If we try to access an index out of the range or use numbers other than an integer, we will get errors.

**Python Strings are Immutable**

In Python, strings are immutable. That means the characters of a string cannot be changed. For example,

message = 'Hola Amigos'

message[0] = 'H'

print(message)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

TypeError: 'str' object does not support item assignment

However, we can assign the variable name to a new string. For example,

message = 'Hola Amigos'

# assign new string to message variable

message = 'Hello Friends'

print(message); # prints "Hello Friends"

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Python Multiline String**

We can also create a multiline string in Python. For this, we use triple double quotes """ or triple single quotes '''. For example,

# multiline string

message = """

Never gonna give you up

Never gonna let you down

"""

print(message)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

Never gonna give you up

Never gonna let you down

In the above example, anything inside the enclosing triple quotes is one multiline string.

**Python String Operations**

Many operations can be performed with strings, which makes it one of the most used [data types](https://www.programiz.com/python-programming/variables-datatypes) in Python.

**1. Compare Two Strings**

We use the == operator to compare two strings. If two strings are equal, the operator returns True. Otherwise, it returns False. For example,

str1 = "Hello, world!"

str2 = "I love Swift."

str3 = "Hello, world!"

# compare str1 and str2

print(str1 == str2)

# compare str1 and str3

print(str1 == str3)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

False

True

In the above example,

1. str1 and str2 are not equal. Hence, the result is False.
2. str1 and str3 are equal. Hence, the result is True.

**2. Join Two or More Strings**

In Python, we can join (concatenate) two or more strings using the + operator.

greet = "Hello, "

name = "Jack"

# using + operator

result = greet + name

print(result)

# Output: Hello, Jack

[Run Code](https://www.programiz.com/python-programming/online-compiler)

In the above example, we have used the + operator to join two strings: greet and name.

**Iterate Through a Python String**

We can iterate through a string using a [for loop](https://www.programiz.com/python-programming/for-loop). For example,

greet = 'Hello'

# iterating through greet string

for letter in greet:

print(letter)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

H

e

l

l

o

**Python String Length**

In Python, we use the [len()](https://www.programiz.com/python-programming/methods/built-in/len) method to find the length of a string. For example,

greet = 'Hello'

# count length of greet string

print(len(greet))

# Output: 5

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**String Membership Test**

We can test if a substring exists within a string or not, using the keyword in.

print('a' in 'program') # True

print('at' not in 'battle') # False

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Methods of Python String**

Besides those mentioned above, there are various [string methods](https://www.programiz.com/python-programming/methods/string) present in Python. Here are some of those methods:

|  |  |
| --- | --- |
| Methods | Description |
| [upper()](https://www.programiz.com/python-programming/methods/string/upper) | Converts the string to uppercase |
| [lower()](https://www.programiz.com/python-programming/methods/string/lower) | Converts the string to lowercase |
| [partition()](https://www.programiz.com/python-programming/methods/string/partition) | Returns a tuple |
| [replace()](https://www.programiz.com/python-programming/methods/string/replace) | Replaces substring inside |
| [find()](https://www.programiz.com/python-programming/methods/string/find) | Returns the index of the first occurrence of substring |
| [rstrip()](https://www.programiz.com/python-programming/methods/string/rstrip) | Removes trailing characters |
| [split()](https://www.programiz.com/python-programming/methods/string/split) | Splits string from left |
| [startswith()](https://www.programiz.com/python-programming/methods/string/startswith) | Checks if string starts with the specified string |
| [isnumeric()](https://www.programiz.com/python-programming/methods/string/isnumeric) | Checks numeric characters |
| [index()](https://www.programiz.com/python-programming/methods/string/index) | Returns index of substring |

**Escape Sequences in Python**

The escape sequence is used to escape some of the characters present inside a string.

Suppose we need to include both a double quote and a single quote inside a string,

example = "He said, "What's there?""

print(example) # throws error

[Run Code](https://www.programiz.com/python-programming/online-compiler)

Since strings are represented by single or double quotes, the compiler will treat "He said, " as a string. Hence, the above code will cause an error.

To solve this issue, we use the escape character \ in Python.

# escape double quotes

example = "He said, \"What's there?\""

# escape single quotes

example = 'He said, "What\'s there?"'

print(example)

# Output: He said, "What's there?"

[Run Code](https://www.programiz.com/python-programming/online-compiler)

Here is a list of all the escape sequences supported by Python.

|  |  |
| --- | --- |
| Escape Sequence | Description |
| \\ | Backslash |
| \' | Single quote |
| \" | Double quote |
| \a | ASCII Bell |
| \b | ASCII Backspace |
| \f | ASCII Formfeed |
| \n | ASCII Linefeed |
| \r | ASCII Carriage Return |
| \t | ASCII Horizontal Tab |
| \v | ASCII Vertical Tab |
| \ooo | Character with octal value ooo |
| \xHH | Character with hexadecimal value HH |

**Python String Formatting (f-Strings)**

Python [f-Strings](https://www.programiz.com/python-programming/string-interpolation#:~:text=f%2Dstrings,Python%20expressions%20inside%20string%20constants.) makes it easy to print values and variables. For example,

name = 'Cathy'

country = 'UK'

print(f'{name} is from {country}')

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

Cathy is from UK

Here, f'{name} is from {country}' is an **f-string**.

This new formatting syntax is powerful and easy to use. From now on, we will use f-Strings to print strings and variables.

**Also Read:**

* [Python str()](https://www.programiz.com/python-programming/methods/built-in/str)
* [Python String Interpolation](https://www.programiz.com/python-programming/string-interpolation)

Before we wrap up, let’s put your knowledge of Python string to the test! Can you solve the following challenge?

Challenge:

Write a function to double every letter in a string.

* For input 'hello', the return value should be 'hheelllloo'.