Strings in Python

In Python, a string is a data type that is typically used to represent text. A string could be any series of characters, including letters, numbers, spaces, etc.

Textual data in Python is handled with **str** objects, more commonly known as **strings**. They are immutable sequences of **Unicode code points**. Unicode code points can represent a character, but can also have other meanings, such as when formatting, for example. Python, unlike other languages, does not have a **char** type, so a single character is rendered simply by a string of length 1.

In most languages (Python included), a string must be enclosed in either single quotes (') or double quotes (") when assigning it to a variable.

Example to create strings

```
a = "Hello World"
b = "My age is 24"
c = "122334"
d = "Good Bye"
print(type(a), a)
print(type(b), b)
print(type(c), c)
print(type(d), d)
```

Output

```
<class 'str'> Hello World
<class 'str'> My age is 24
<class 'str'> 122334
<class 'str'> Good Bye
```

Difference in a Number and a String

If you need to store numbers that may have calculations performed on them, do not make them a string. There is a difference between the following two declarations:

```
a = 10
b = "10"
```

In the first line, the value 10 is a number. In the second line, it is a string.

Including numbers in a string simply makes them part of that string. So, you cannot do things like add two numbers together if they are actually a string. Arithmetic operators have a different purpose when used on strings. For example, when working with numbers, the plus sign (+) adds two numbers together, but when working with strings, it concatenates the strings together.

```
print(10 + 20)
print("10" + "20")
```

Output

```
30
1020
```

The first line is a result of **adding two numbers**. The second line is a result of **concatenating two strings**.

Multiline Strings

You can assign a multiline string to a variable by using three quotes:

```
a = """This is a multiline string
You can write whatever you want to
Python is the best language ever made
"""

print(a)
```

Output

```
This is a multiline string
You can write whatever you want to
Python is the best language ever made
```

Accessing a single character from a String & Slicing String

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.

Just as same as List, we can access element through index and even slice it as we do in lists.

Example

```
a = "We are learning Python Language"

print(a)
print(a[0])
print(a[2])
print(a[-1])
print(a[0:5])
print(a[5:10])
print(a[-1:-10:-1])
print(a[-1::-1])
```

Output

```
We are learning Python Language
W

e
We ar
e lea
egaugnaL
egaugnaL nohtyP gninrael era eW
```

String Built-In Methods

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

Method	Description
capitalize()	Converts the first character to upper case
count()	Returns the number of times a specified value occurs in a string
startswith()	Returns true if the string starts with the specified value
endswith()	Returns true if the string ends with the specified value
find()	Searches the string for a specified value and returns the position of where it was found
index()	Searches the string for a specified value and returns the position of where it was found
isalnum()	Returns True if all characters in the string are alphanumeric
isalpha()	Returns True if all characters in the string are in the alphabet
isdigit()	Returns True if all characters in the string are digits
lower()	Converts a string into lower case
upper()	Converts a string into upper case
islower()	Returns True if all characters in the string are lower case
isupper()	Returns True if all characters in the string are upper case
replace()	Returns a string where a specified value is replaced with a specified value
split()	Splits the string at the specified separator, and returns a list
swapcase()	Swaps cases, lower case becomes upper case and vice versa
title()	Converts the first character of each word to upper case

Python string capitalize() method

Converts the first character to upper case.

```
x = "hello python developer"
print(x.capitalize())

y = "Hello Python developer"
print(y.capitalize())

# Nothing will happen if first character is not a alphabet
z = "33 is my age"
print(z.capitalize())
```

Output

```
Hello python developer
Hello python developer
33 is my age
```

Python string count() method

Returns the number of times a specified value occurs in a string.

```
x = "We are learning python language"

print(x.count("a"))
print(x.count("e"))
print(x.count("pyt"))
print(x.count("ing"))
print(x.count("z"))
print(x.count("ares"))
```

Output

```
4
4
1
1
0
0
```

Python string startswith() method

Returns true if the string starts with the specified value.

```
x = "We are learning python language"

print(x.startswith("W"))
print(x.startswith("w"))
print(x.startswith("We"))
print(x.startswith("Well"))
print(x.startswith("xyz"))
```

Output

```
True
False
True
False
False
```

Python string endswith() method

Returns true if the string ends with the specified value.

```
x = "We are learning python language"

print(x.endswith("e"))
print(x.endswith("age"))
print(x.endswith("lang"))
print(x.endswith("language"))
print(x.endswith("xyz"))
```

Output

```
True
True
False
True
False
```

Python string find() method

Searches the string for a specified value and returns the position of where it was found.

```
x = "We are learning python language"

print(x.find("W"))
print(x.find("l"))
print(x.find("python"))

# If character does not exist, it will return -1
print(x.find("z"))
```

Output

```
0
7
16
-1
```

Python string index() method

Searches the string for a specified value and returns the position of where it was found.

```
x = "We are learning python language"

print(x.index("W"))
print(x.index("l"))
print(x.index("python"))

# If character does not exist, it will give us an error
print(x.index("z"))
```

Output

```
0
7
16
Traceback (most recent call last):
  File "C:\Users\aniru\Desktop\Python Programming\strings.py", line 8, in <module>
    print(x.index("z"))
ValueError: substring not found
```

Python string isalnum() method

Returns True if all characters in the string are alphanumeric.

```
x = "We are learning python language"
y = "My age is 24"
a = "thishasonlynumbers234andletters"
b = "hishasnospace$#$%"
print(x.isalnum())
print(y.isalnum())
print(a.isalnum())
print(b.isalnum())
```

Output

```
False
False
True
False
```

Python string isalpha() method

Returns True if all characters in the string are in the alphabet.

```
x = "We are learning python language"
y = "My age is 24"
a = "thishasonlyletters"
b = "hishasnospace$#$%"
print(x.isalpha())
print(y.isalpha())
print(a.isalpha())
print(b.isalpha())
```

Output

```
False
False
True
False
```

Python string isdigit() method

Returns True if all characters in the string are digits.

```
x = "We are learning python language"
y = "32432432"
a = "432dadwa321"
b = "32 343 43 34 343"
print(x.isdigit())
print(y.isdigit())
print(a.isdigit())
print(b.isdigit())
```

Output

```
False
True
False
False
```

Python string lower() method

Converts a string into lower case.

```
a = "We are learning python language"
b = "THIS IS A DUMMY TEXT"
c = "PYTHON is FAst"
d = "1023945ABCDE"
print(a.lower())
print(b.lower())
print(c.lower())
print(d.lower())
```

Output

```
we are learning python language
this is a dummy text
python is fast
1023945abcde
```

Python string upper() method

Converts a string into upper case.

```
a = "We are learning python language"
b = "THIS IS A DUMMY TEXT"
c = "PYTHON is FAst"
d = "1023945abcde$##@!"
print(a.upper())
print(b.upper())
print(c.upper())
print(d.upper())
```

Output

```
WE ARE LEARNING PYTHON LANGUAGE
THIS IS A DUMMY TEXT
PYTHON IS FAST
1023945ABCDE$##@!
```

Python string islower() and isupper() method

```
a = "We are learning python language"
b = "THIS IS A DUMMY TEXT"
print(a.islower())
print(a.isupper())
print(b.islower())
print(b.isupper())
```

Output

```
False
False
False
True
```

Python string replace() method

Returns a string where a specified value is replaced with a specified value.

```
a = "We are learning python language"
print(a)

b = a.replace("a", "z")
print(b)
```

Output

```
We are learning python language
We zre lezrning python lznguzge
```

Python string swapcase() method

Swaps cases, lower case becomes upper case and vice versa.

```
a = "We are learning python language"
b = "HEEllo WorLD"
print(a.swapcase())
print(b.swapcase())
```

Output

```
wE ARE LEARNING PYTHON LANGUAGE
heeLLO wORld
```

Python string split() method

Splits the string at the specified separator, and returns a list.

```
x = "We are learning python language"

print(x.split())
print(x.split("a"))
```

Output

```
['We', 'are', 'learning', 'python', 'language']
['We ', 're le', 'rning python l', 'ngu', 'ge']
```

Practice Examples (String)

1. Python program to print even length words in a string.
2. Write a Python program to calculate the length of a string entered by a User.
3. Write a Python program to reverse a string entered by User.
4. Count the number of vowels in a string entered by User.
5. Remove the even index characters from the string entered by User.
   String: “PythonisGreat”
   Answer: “PtoGet”
6. Write a Python program to reverse a string if its length is a multiple of 4.
7. Count number of digits in a string entered by user.
8. Check if the String entered by user is palindrome or not.
9. Find all duplicate characters in string entered by user.
10. Reverse Sort a String entered by user.

For solutions, check the end of the book.