Course	
Term	
Week	
Date	
Chapter. Topic	7. Lists and Tuples

Python Lists: String processing

Siva R Jasthi

Computer Science and Cybersecurity

Metropolitan State University

Lists

List is a collection which is **ordered** and **changeable**. Allows **duplicate** members.

Lists: An introduction
https://www.w3schools.com/python/python_lists.asp

Lists: An introduction
https://openbookproject.net/thinkcs/python/english3e/lists.html

List Methods
http://www.python-ds.com/python-3-list-methods

Built-in Functions
https://docs.python.org/3/library/functions.html

List vs String

- Traversal (for loop)
- Membership operator (in operator)
- Index access [0 to n-1]
- Split function   (string.split() will give a list)

Overview

Python strings and lists are similar but different.

Similar: syntax, access mechanisms, operators (+, in, for)

Different:

- Strings are **immutable**, lists are **mutable**
- Strings are homogeneous, lists are heterogeneous
- myString[1] is a string, myList[1] might not be a list

Use the subscript notation

```
myList = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
myString = "abcdefghijklmnopqrstuvwxyz"
```

Use the `len( )` function to get the size.

```
myList[-1] == 9

myString[-2] ==  'y'
```

Use the subscript [ ] to get elements

```
myList = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
myString = "abcdefghijklmnopqrstuvwxyz"
```

Use the `[]` to get components.

`myList[3]` gives 3

`myString[3]` gives "d"

`myList[-1]` gives 9

`myString[-2]` gives "y“

myList[-3] gives 7

myString[-3] gives x

Getting the length

```
myList = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
myString = "abcdefghijklmnopqrstuvwxyz"
```

Use the `len( )` function to get the size.

```
len(myList) == 10

len(myString) == 26
```

Empty list and empty string

Empty list and empty string:

```
myList = []
myString = ""
```

Use slicing [:] to get sub-lists and sub-strings

```
myList = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
myString = "abcdefghijklmnopqrstuvwxyz"
```

Use the `[:]` to get sub-lists and substrings.

`myList[3:7]` gives `[3, 4, 5, 6]`

`myString[3:7]` gives `"defg"`

This is the *slicing* operator.

Slicing the strings and lists (examples)

```
myList = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
myString = "abcdefghijklmnopqrstuvwxyz"
```

More slicing tricks.

`myList[:7]` gives `[0,1, 2, 3, 4, 5, 6]`

`myString[:7]` gives `"abcdefg"`

`myList[-3:]` gives `[7, 8, 9]`

`myString[-3:]` gives `"xyz"`

Concatenation: Use + to add lists and strings

```
L1 = [0, 1, 2, 3 ]
L2 = [ 4, 5]
str1 = "abcdefghij"
str2 = "klmno"
```

Use + operator for concatenation.

`L1 + L2` gives `[0,1, 2, 3, 4, 5]`

`str1 + str2` gives `"abcdefghijklmno"`

Membership operator (in)

```
L1 = [0, 1, 2, 3 ]
str1 = “python"
```

Use `in` operator for checking the presence of an element.

```
(1 in L1) is True
(99 not in L1) is True
(88 in L1) is False
(3 not in L1) is False

‘p’ in str1 is True
‘y’ not in str1 is False
‘j’ in str1 is False
‘n’ not in str1 is False
```

Lists are mutable (can be changed)

```
myList = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
myList[4] = 44
```

changes myList to become:

```
[0, 1, 2, 3, 44, 5, 6, 7, 8, 9]
```

Lists are mutable (example 2)

```
myList = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
myList[4:7] = [11, 22]
```

changes myList to become:

```
myList = [0, 1, 2, 3, 11, 22, 7, 8, 9]
```

Strings are immutable (can not be changed)

```
   myString = "abcdefghijklmnopqrstuvwxyz"
   myString[4] = "X"
```

is an error

Strings are homogeneous

```
myString = "abcdefghijklmnopqrstuvwxyz"
```

each `myString[i]` has the same type, a string.

Lists can be heterogeneous (ex.1)

```
myList= [ "car", 5, True, 76.2 ]
```

is perfectly legal.

`myList[0]`	is a string
`myList[1]`	is an int
`myList[2]`	is a boolean
`myList[3]`	is a float

Lists can be heterogeneous (ex.2)

```
myList= ["car", 5, [True, "blue"], 76.2]
```

is perfectly legal.

`myList[0]`	is a string
`myList[1]`	is an int
`myList[2]`	is a list
`myList[3]`	is a float

A sting element is always a string

```
myList = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
myString = "abcdefghijklmnopqrstuvwxyz"
```

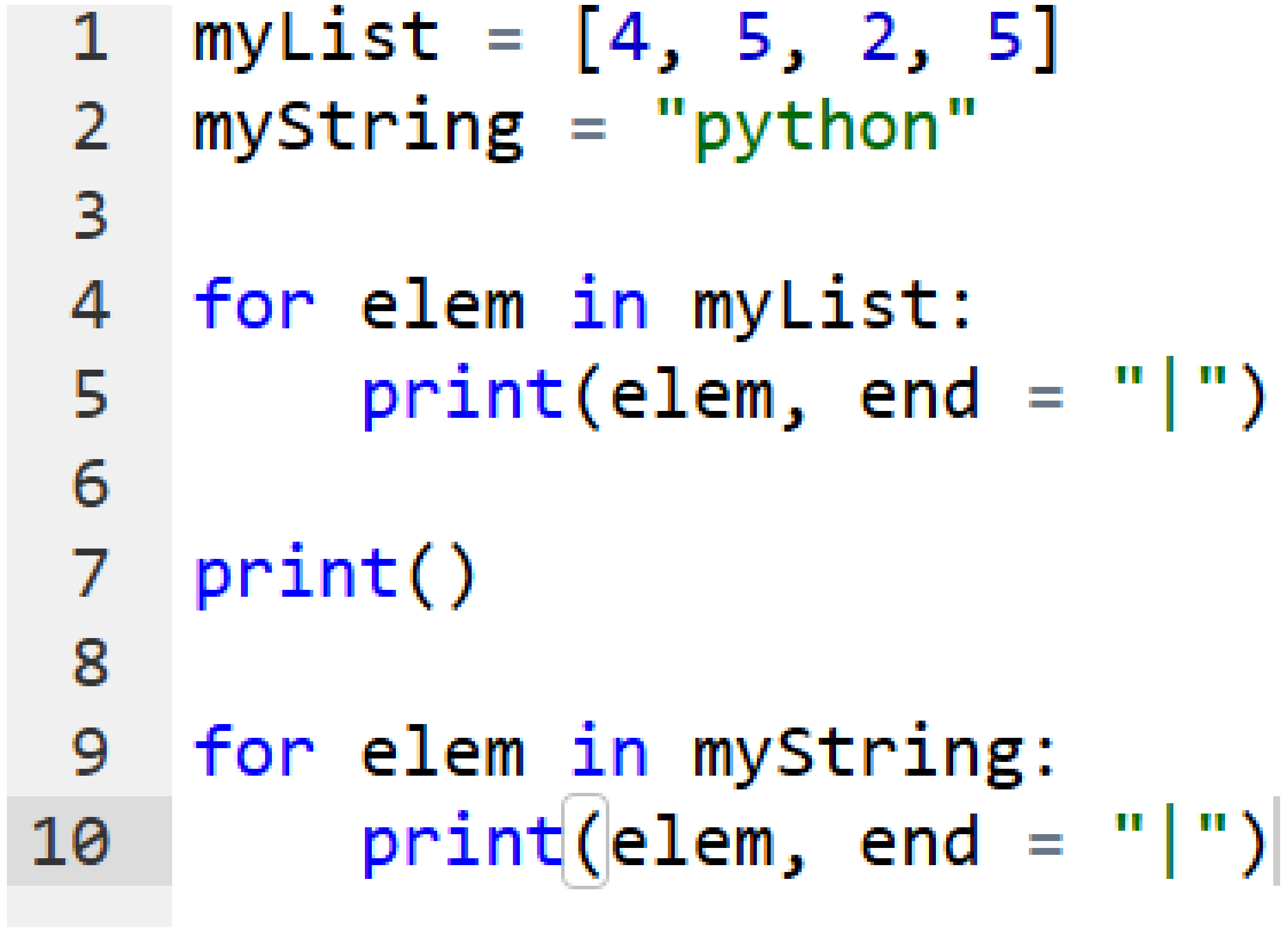
Then,

`myList[7]` is an integer

`myString[7]` is a string "h"

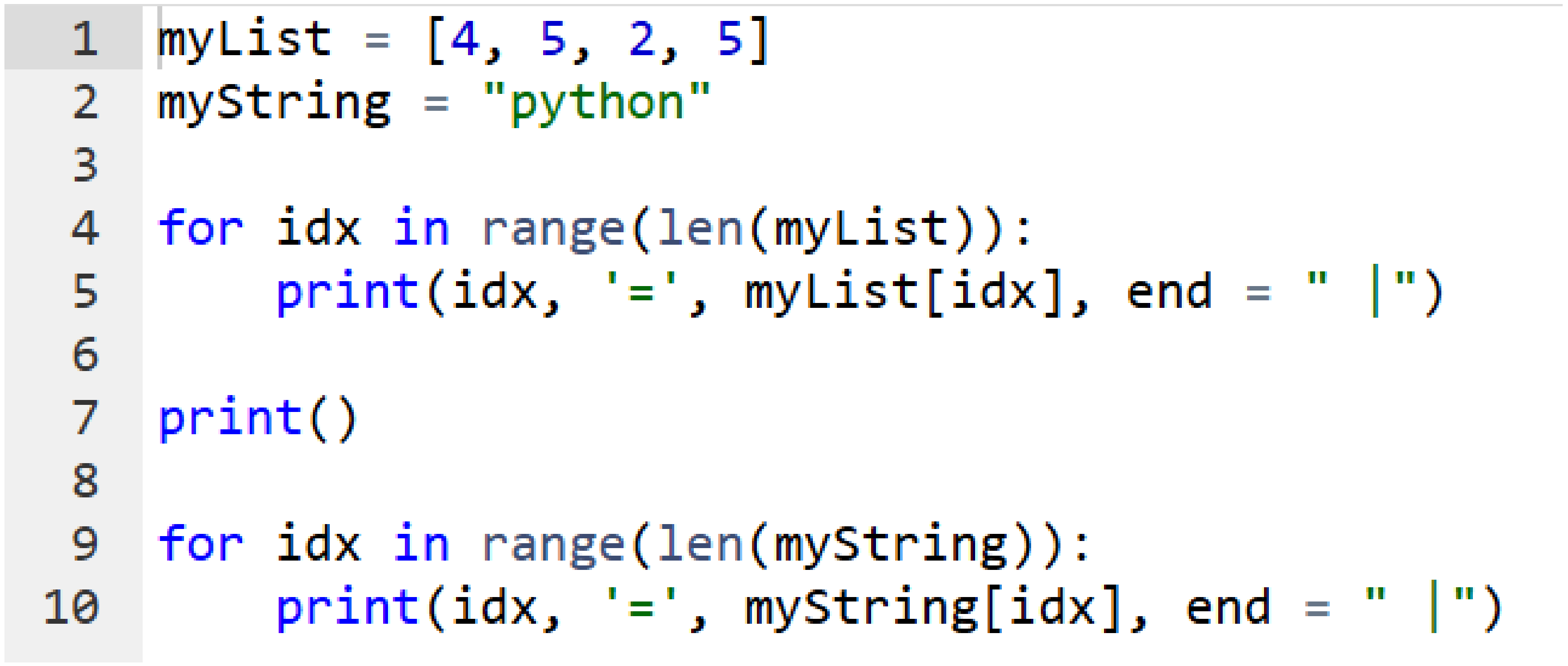
`myString` and `myString[7]` are both strings.

Traversals: for loop (method 1)



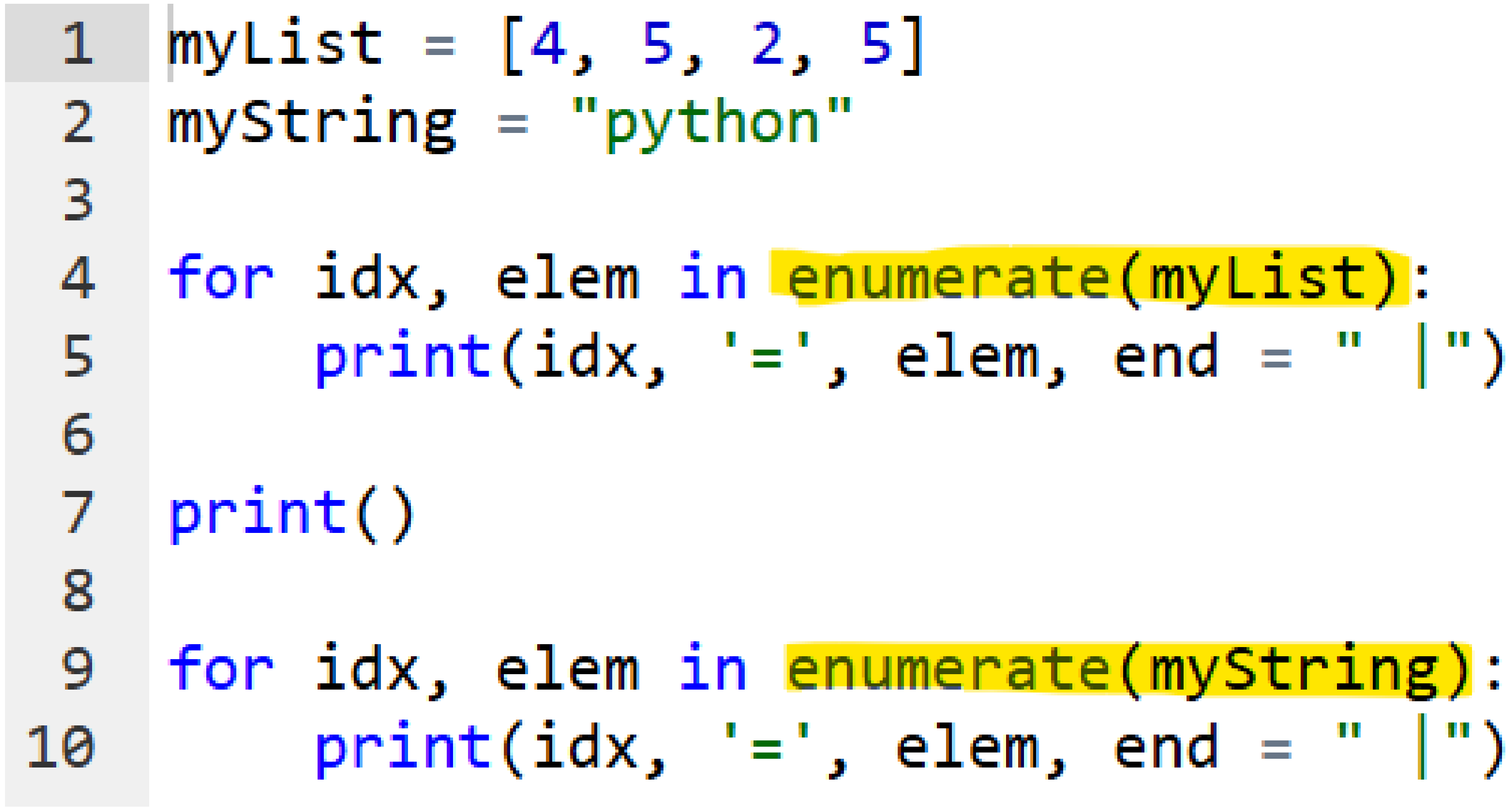
```python
myList = [4, 5, 2, 5]
myString = "python"

for elem in myList:
    print(elem, end = "|")

print()

for elem in myString:
    print(elem, end = "|")
```

Traversals: for loop (method 2)



```python
myList = [4, 5, 2, 5]
myString = "python"

for idx in range(len(myList)):
    print(idx, '=', myList[idx], end = " |")

print()

for idx in range(len(myString)):
    print(idx, '=', myString[idx], end = " |")
```

Traversals: for loop (method 3)



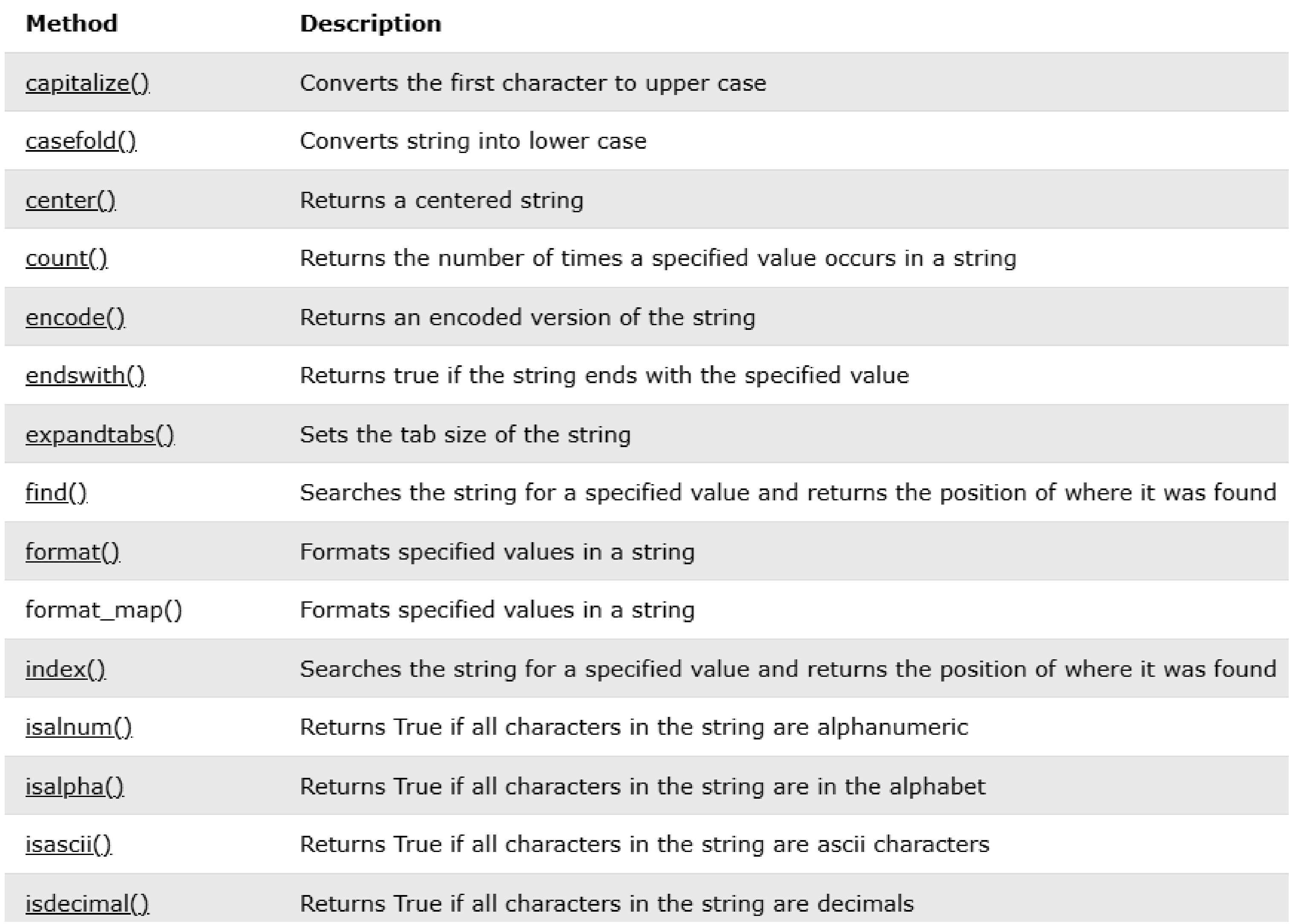
```python
myList = [4, 5, 2, 5]
myString = "python"

for idx, elem in enumerate(myList):
    print(idx, '=', elem, end = " |")

print()

for idx, elem in enumerate(myString):
    print(idx, '=', elem, end = " |")
```

List Methods list.method_name(params)

Method	Purpose
append(x)	Add x to the end of the list
extend(list_x)	Add all items from list_x at the end of the list
insert(i,x)	Inserts an item at a given position. The first argument is the index of the element before which to insert. For example, a.insert(0, x) inserts at the front of the list.
remove(x)	Removes the first item x (note: there can be multiple items x in the list)
pop()	Removes the last item and returns the item
clear( )	Removed all elements in the list. Empties the list.
index(x)	Returns the index of the first item x.
count(x)	Counts the number of times x is appearing in the list
sort( )	Sorts the elements in ascending order. sort(reverse=True) sorts the elements in descending order
reverse( )	Reverses a list
copy( )	Returns a copy of the list. You can also use "list" built-in function for the same purpose.

https://www.w3schools.com/python/python_ref_list.asp

String Methods str1.method_name(params)

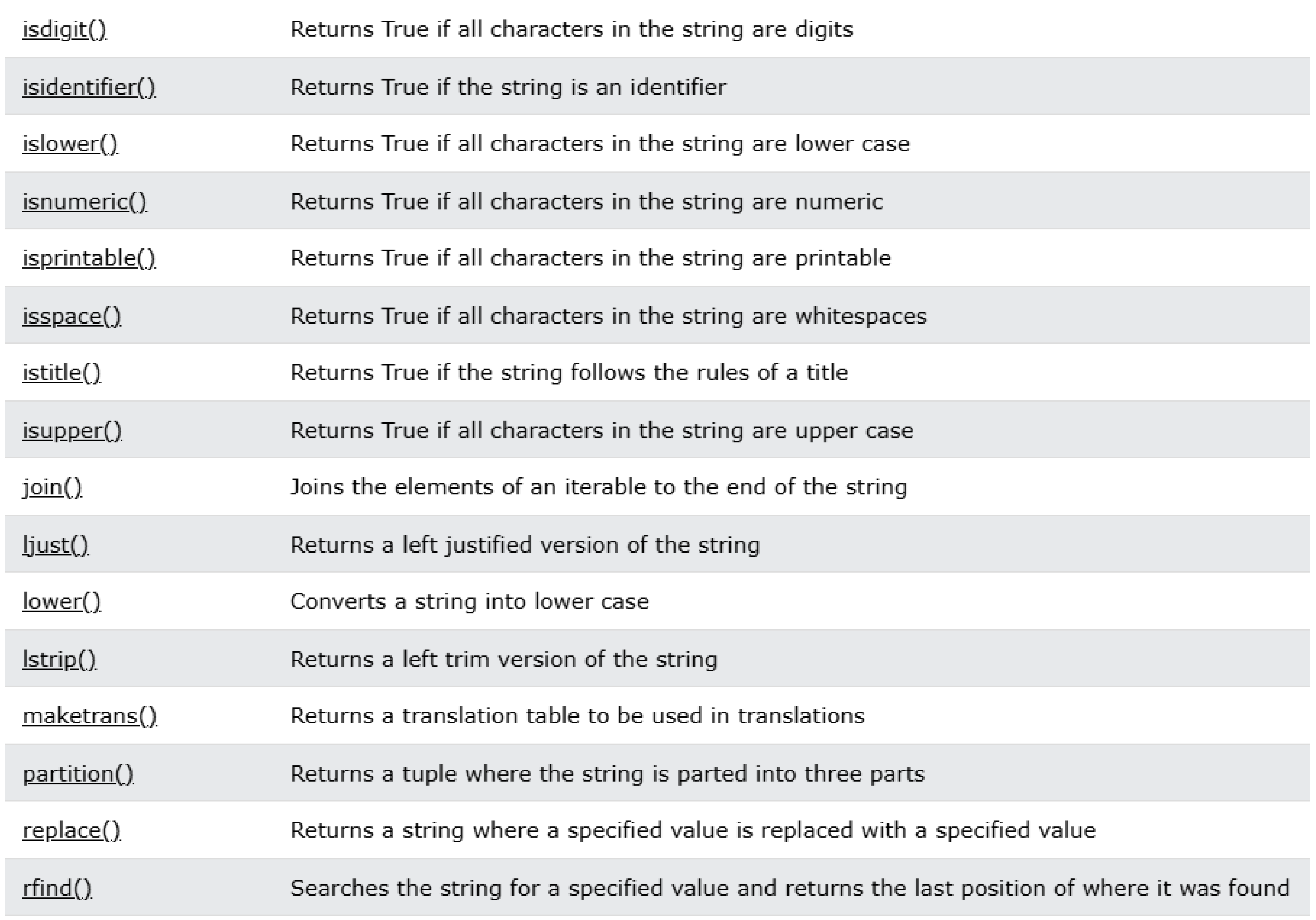


Method	Description
capitalize()	Converts the first character to upper case
casefold()	Converts string into lower case
center()	Returns a centered string
count()	Returns the number of times a specified value occurs in a string
encode()	Returns an encoded version of the string
endswith()	Returns true if the string ends with the specified value
expandtabs()	Sets the tab size of the string
find()	Searches the string for a specified value and returns the position of where it was found
format()	Formats specified values in a string
format_map()	Formats specified values in a string
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
isascii()	Returns True if all characters in the string are ascii characters
isdecimal()	Returns True if all characters in the string are decimals

https://www.w3schools.com/python/python_strings_methods.asp

Note: All string methods return new values. They do not change the original string.

String Methods (Contd.)

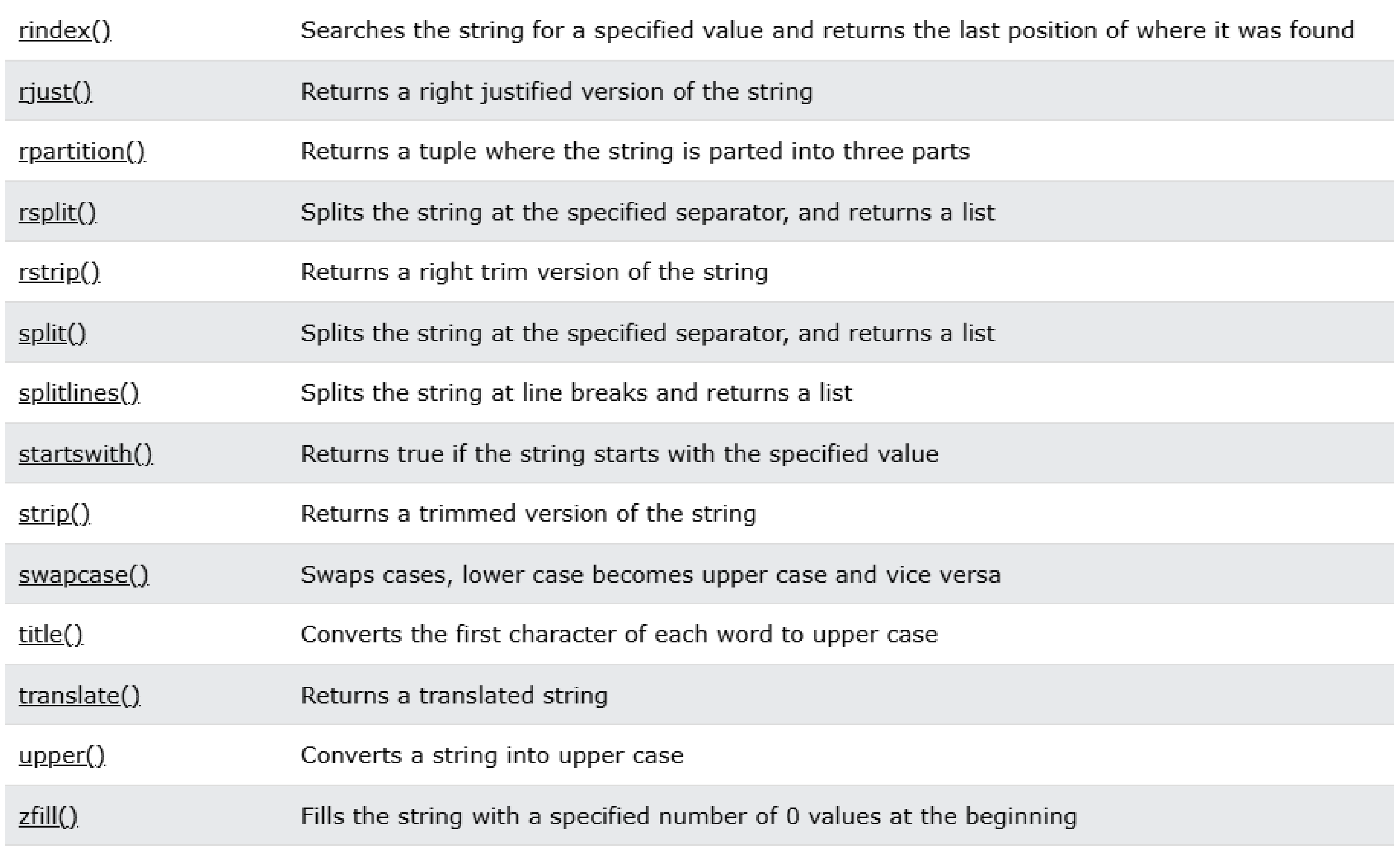


isdigit()	Returns True if all characters in the string are digits
isidentifier()	Returns True if the string is an identifier
islower()	Returns True if all characters in the string are lower case
isnumeric()	Returns True if all characters in the string are numeric
isprintable()	Returns True if all characters in the string are printable
isspace()	Returns True if all characters in the string are whitespaces
istitle()	Returns True if the string follows the rules of a title
isupper()	Returns True if all characters in the string are upper case
join()	Joins the elements of an iterable to the end of the string
ljust()	Returns a left justified version of the string
lower()	Converts a string into lower case
lstrip()	Returns a left trim version of the string
maketrans()	Returns a translation table to be used in translations
partition()	Returns a tuple where the string is parted into three parts
replace()	Returns a string where a specified value is replaced with a specified value
rfind()	Searches the string for a specified value and returns the last position of where it was found

https://www.w3schools.com/python/python_strings_methods.asp

Note: All string methods return new values. They do not change the original string.

String Methods (Contd.)



rindex()	Searches the string for a specified value and returns the last position of where it was found
rjust()	Returns a right justified version of the string
rpartition()	Returns a tuple where the string is parted into three parts
rsplit()	Splits the string at the specified separator, and returns a list
rstrip()	Returns a right trim version of the string
split()	Splits the string at the specified separator, and returns a list
splitlines()	Splits the string at line breaks and returns a list
startswith()	Returns true if the string starts with the specified value
strip()	Returns a trimmed version of the string
swapcase()	Swaps cases, lower case becomes upper case and vice versa
title()	Converts the first character of each word to upper case
translate()	Returns a translated string
upper()	Converts a string into upper case
zfill()	Fills the string with a specified number of 0 values at the beginning

https://www.w3schools.com/python/python_strings_methods.asp

Note: All string methods return new values. They do not change the original string.

Lists and Strings: Comparison summary

Feature	Lists	Strings
Definition	Ordered and changeable collection of elements	Ordered and immutable sequence of characters
Mutability	Mutable (can be modified)	Immutable (cannot be modified)
Data Type	Can contain different data types (heterogeneous)	Contains only characters (homogeneous)
Indexing	Supports indexing (myList[i])	Supports indexing (myString[i])
Slicing	Supports slicing (myList[start:end])	Supports slicing (myString[start:end])
Membership (in operator)	Can check if an element exists in the list	Can check if a character or substring exists
Concatenation (+ operator)	Joins two lists ([1, 2] + [3, 4] → [1, 2, 3, 4])	Joins two strings ("abc" + "def" → "abcdef")
Length (len() function)	Returns the number of elements in the list	Returns the number of characters in the string
Modification	Elements can be changed (myList[i] = new_value)	Cannot change characters (myString[i] = new_value gives an error)
Empty Structure	myList = []	myString = ""
Methods	Has methods like .append(), .remove(), .sort(), etc.	Has methods like .upper(), .lower(), .split(), etc.
Conversion	Can convert a string to a list using split()	Can convert a list to a string using join()
Nested Structure	Can contain other lists ([1, "abc", [3, 4]])	Cannot contain other data structures
Traversal	Can be traversed using a for loop	Can be traversed using a for loop