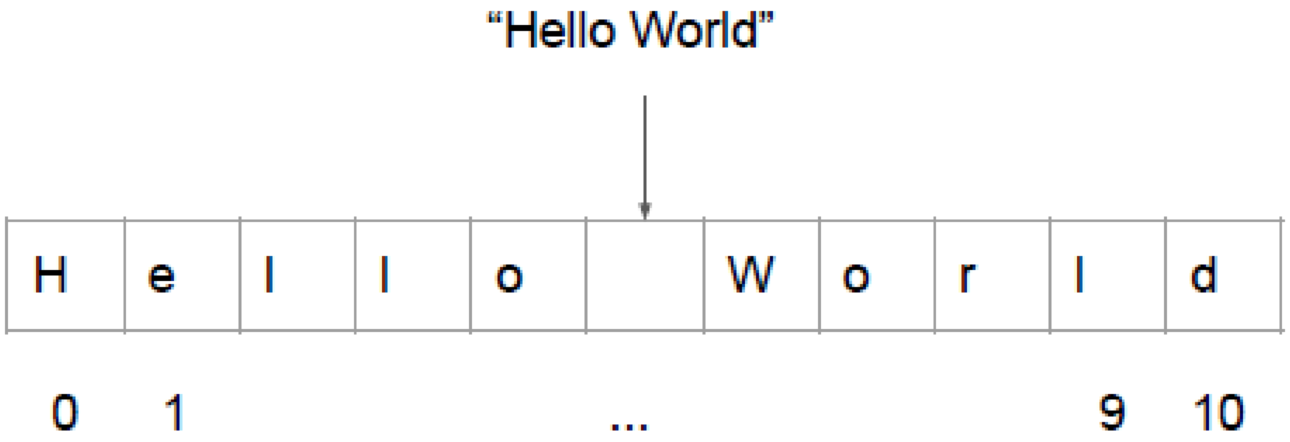
Strings

Spring 2021

Strings

- Strings in Python can be treated as lists, in a way
- Each individual character is its own index
    - This does include white space

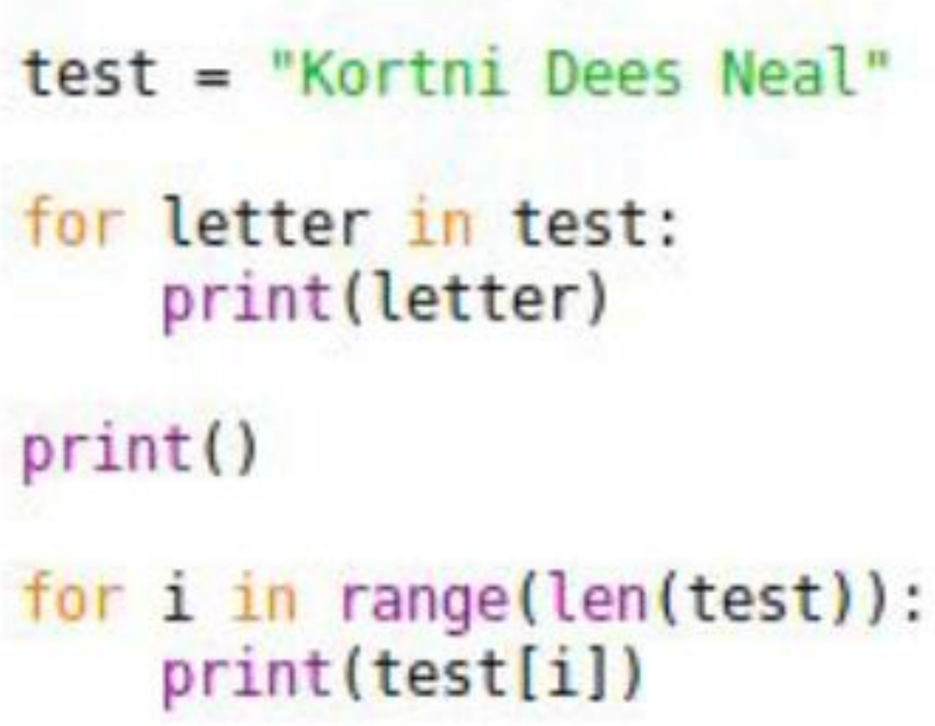


“Hello World”

H	e	l	l	o		W	o	r	l	d
0	1				...				9	10

Loops

- Because strings can be treated as lists, loops that print lists can directly be used to print each individual character within a string
- The len function to find the length of a string also works

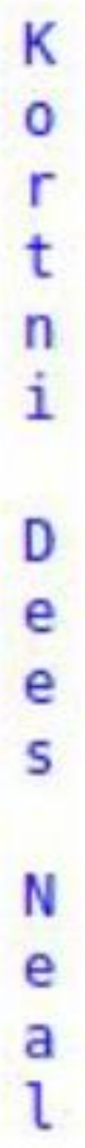


```
test = "Kortni Dees Neal"

for letter in test:
    print(letter)

print()

for i in range(len(test)):
    print(test[i])
```



```
K
o
r
t
n
i
 
D
e
e
s
 
N
e
a
l
```

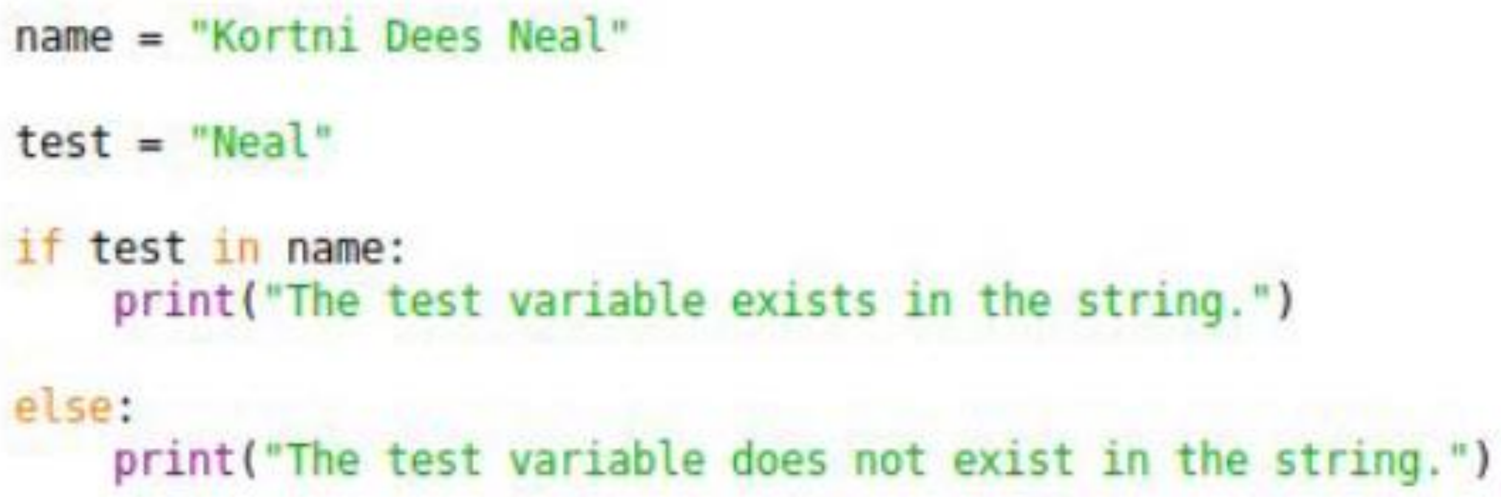
Characters

- Can reference a specific character in a string
- Can also create a substring via same methodology as lists
- *Cannot* change a value of a specific index

message = “Hello World”	
print(message[0])	H
print(message[3])	l
print(message[3:6])	“lo “ (includes space)
print(message[6:])	World
print(message[:5])	Hello

Searching Strings

- Can also search a string via same method of lists



```python
name = "Kortni Dees Neal"

test = "Neal"

if test in name:
    print("The test variable exists in the string.")

else:
    print("The test variable does not exist in the string.")
```

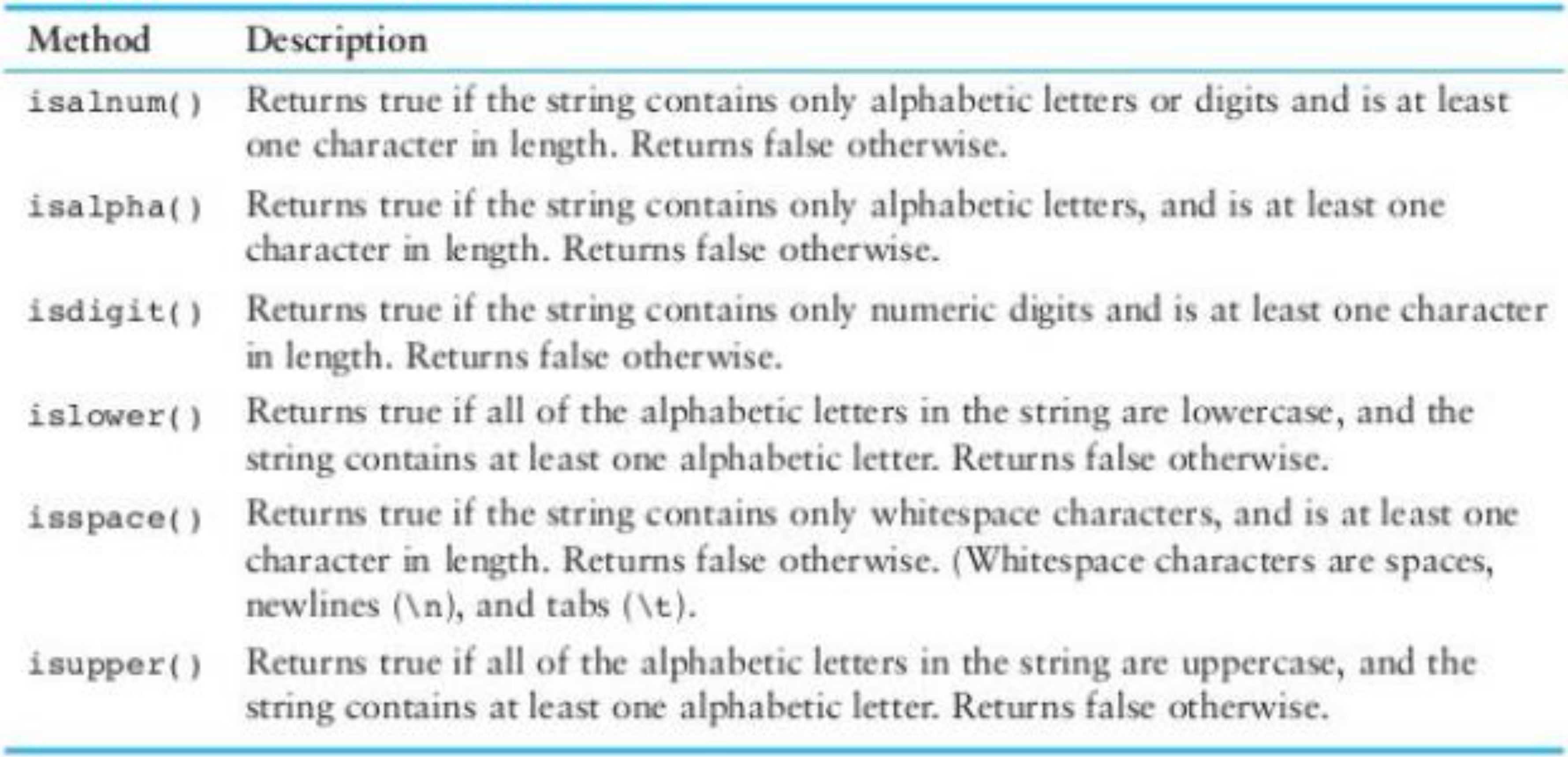
String Splitting

- Strings can be split on a specified character to create a list
  - Character split on is removed entirely
- Done via .split() function
  - Defaults to space if nothing is specified
- Can be print or assigned to a variable

message = “Hello, my name is Kortni”	
print(message.split(“ “))	[“Hello,”, “my”, “name”, “is”, “Kortni”]
print(message.split())	[“Hello,”, “my”, “name”, “is”, “Kortni”]
print(message.split(“,”))	[“Hello”, “ my name is Kortni”]
print(message.split(“m”))	[“Hello, “, “y na”, “e is Kortni”]

String Testing

Table 9-1 Some string testing methods



Method	Description
`isalnum()`	Returns true if the string contains only alphabetic letters or digits and is at least one character in length. Returns false otherwise.
`isalpha()`	Returns true if the string contains only alphabetic letters, and is at least one character in length. Returns false otherwise.
`isdigit()`	Returns true if the string contains only numeric digits and is at least one character in length. Returns false otherwise.
`islower()`	Returns true if all of the alphabetic letters in the string are lowercase, and the string contains at least one alphabetic letter. Returns false otherwise.
`isspace()`	Returns true if the string contains only whitespace characters, and is at least one character in length. Returns false otherwise. (Whitespace characters are spaces, newlines (`\n`), and tabs (`\t`).
`isupper()`	Returns true if all of the alphabetic letters in the string are uppercase, and the string contains at least one alphabetic letter. Returns false otherwise.

String Modification

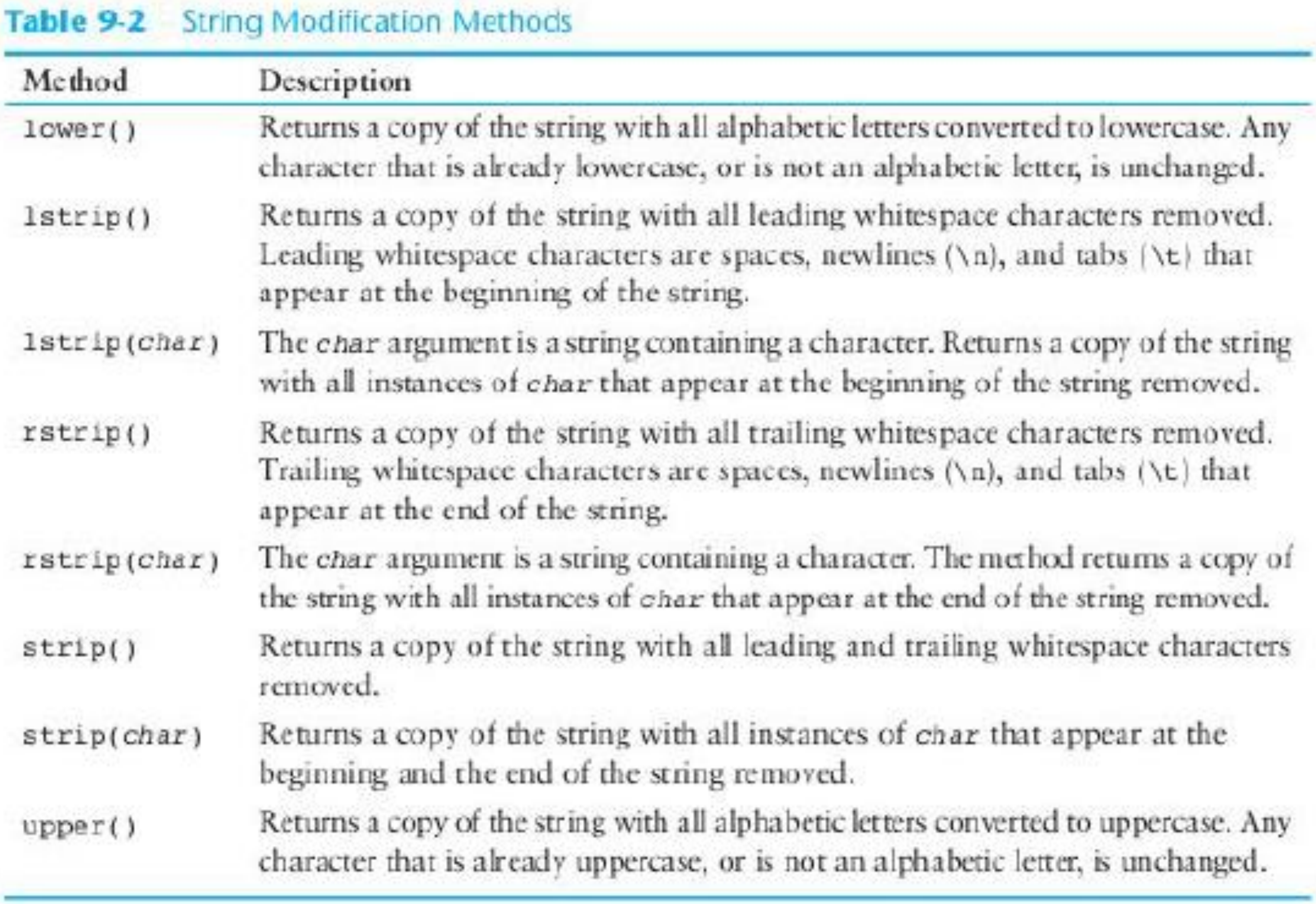


Table 9-2 String Modification Methods

Method	Description
`lower()`	Returns a copy of the string with all alphabetic letters converted to lowercase. Any character that is already lowercase, or is not an alphabetic letter, is unchanged.
`lstrip()`	Returns a copy of the string with all leading whitespace characters removed. Leading whitespace characters are spaces, newlines (`\n`), and tabs (`\t`) that appear at the beginning of the string.
`lstrip(`*`char`*`)`	The *`char`* argument is a string containing a character. Returns a copy of the string with all instances of *`char`* that appear at the beginning of the string removed.
`rstrip()`	Returns a copy of the string with all trailing whitespace characters removed. Trailing whitespace characters are spaces, newlines (`\n`), and tabs (`\t`) that appear at the end of the string.
`rstrip(`*`char`*`)`	The *`char`* argument is a string containing a character. The method returns a copy of the string with all instances of *`char`* that appear at the end of the string removed.
`strip()`	Returns a copy of the string with all leading and trailing whitespace characters removed.
`strip(`*`char`*`)`	Returns a copy of the string with all instances of *`char`* that appear at the beginning and the end of the string removed.
`upper()`	Returns a copy of the string with all alphabetic letters converted to uppercase. Any character that is already uppercase, or is not an alphabetic letter, is unchanged.

String Search/Replace

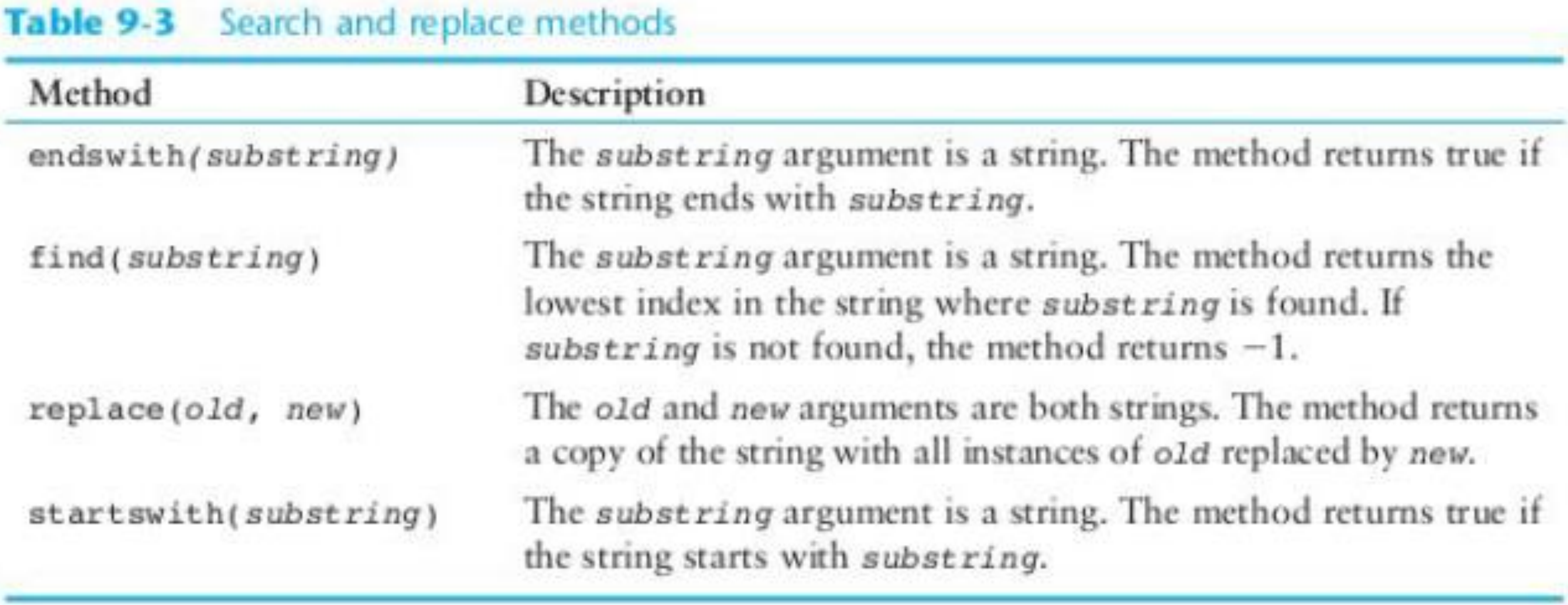
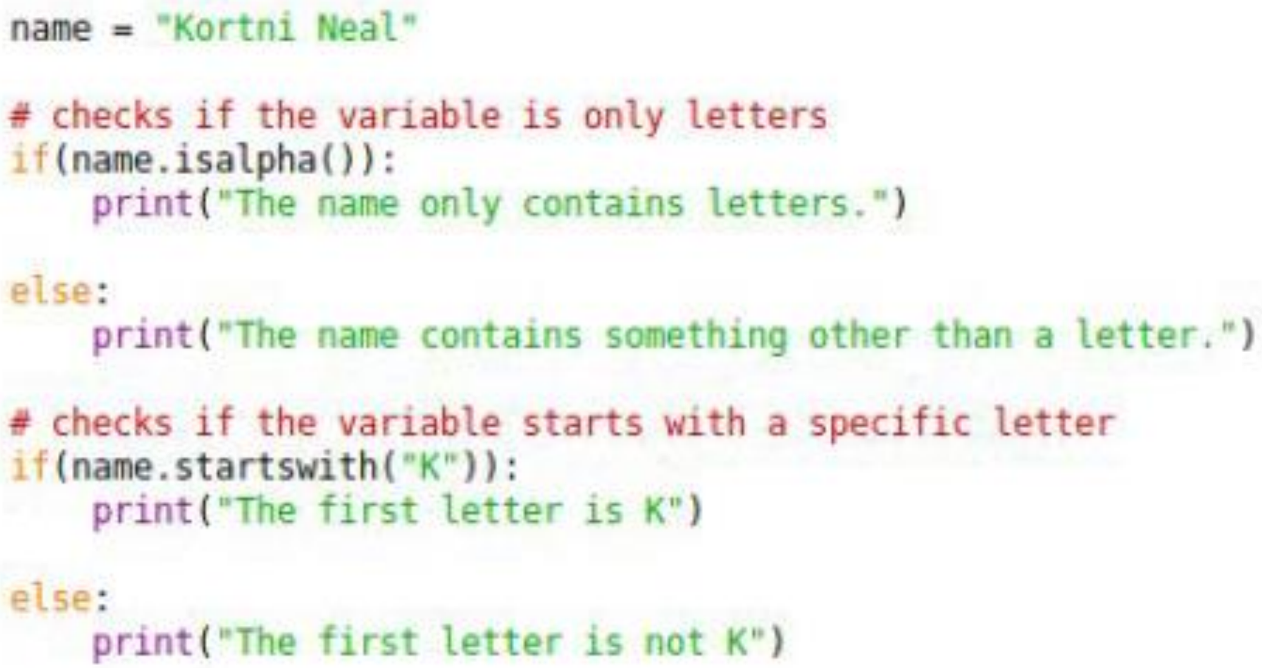


Table 9-3 Search and replace methods

Method	Description
endswith(*substring*)	The *substring* argument is a string. The method returns true if the string ends with *substring*.
find(*substring*)	The *substring* argument is a string. The method returns the lowest index in the string where *substring* is found. If *substring* is not found, the method returns −1.
replace(*old*, *new*)	The *old* and *new* arguments are both strings. The method returns a copy of the string with all instances of *old* replaced by *new*.
startswith(*substring*)	The *substring* argument is a string. The method returns true if the string starts with *substring*.

String Examples



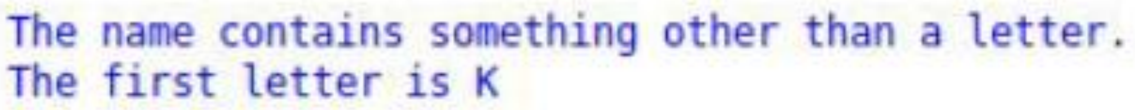
```python
name = "Kortni Neal"

# checks if the variable is only letters
if(name.isalpha()):
    print("The name only contains letters.")

else:
    print("The name contains something other than a letter.")

# checks if the variable starts with a specific letter
if(name.startswith("K")):
    print("The first letter is K")

else:
    print("The first letter is not K")
```



```
The name contains something other than a letter.
The first letter is K
```