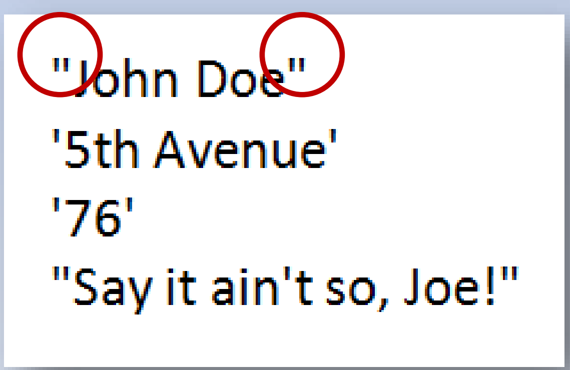
Strings

- Sequence of characters that is treated as a single item
- Written as a sequence of characters surrounded by either single quotes (') or double quotes (").

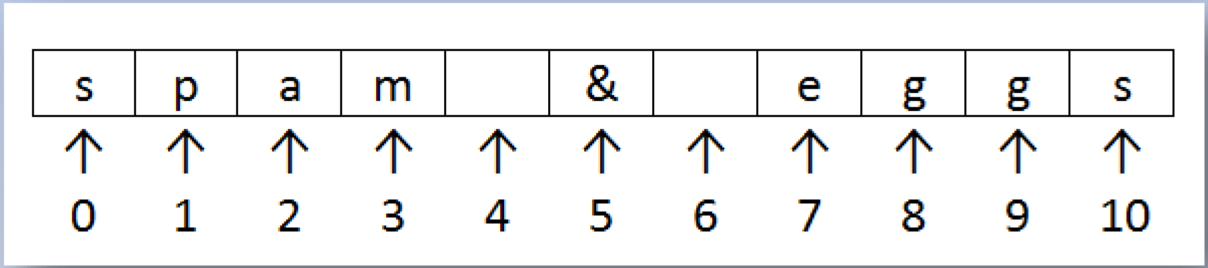


```
"John Doe"
'5th Avenue'
'76'
"Say it ain't so, Joe!"
```

Opening and closing quotation marks must be the same type

Indices and Slices

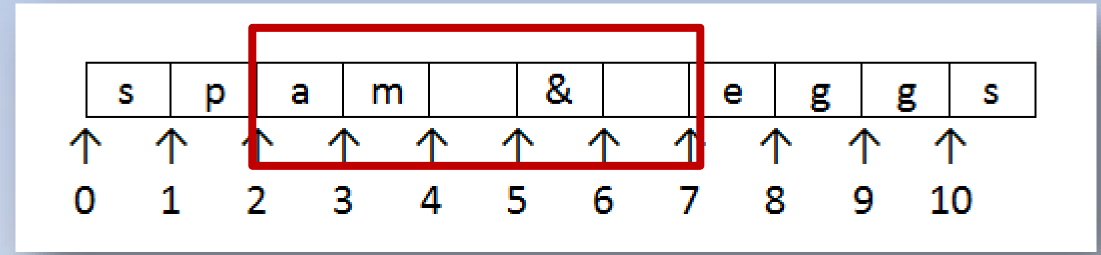
- Position or index of a character in a string
  - Identified with one of the numbers 0, 1, 2, 3, ...



Indices of the characters of the string "spam & eggs".

Indices and Slices

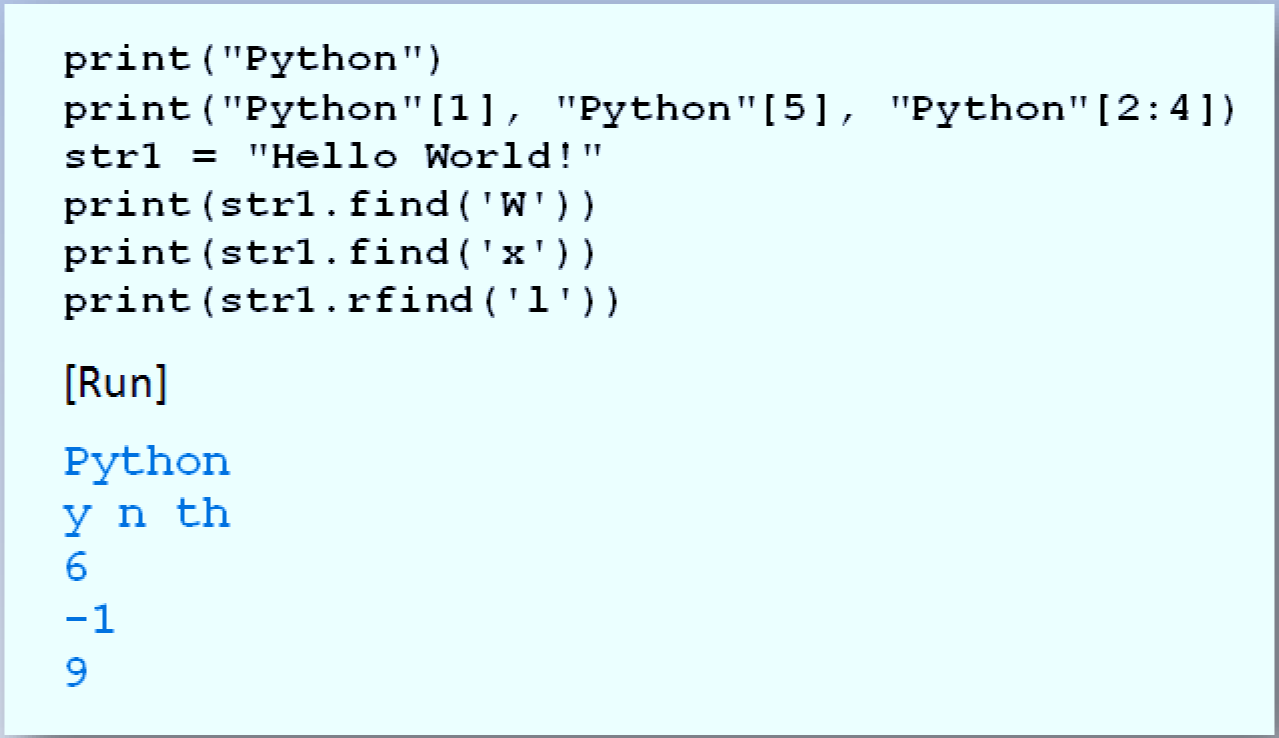
- If str1 is a string, then **`str1[m:n]`** is the substring beginning at position m and ending at position n - 1
  - Example “spam & eggs”[2:7]



Aid to visualizing slices.

Indices and Slices

- Example: Program shows use of indices



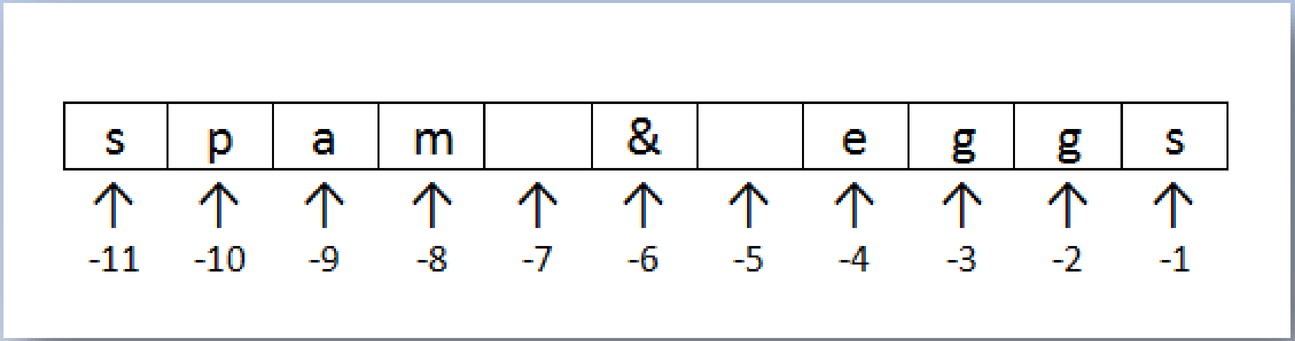
```
print("Python")
print("Python"[1], "Python"[5], "Python"[2:4])
str1 = "Hello World!"
print(str1.find('W'))
print(str1.find('x'))
print(str1.rfind('l'))
```

[Run]

```
Python
y n th
6
-1
9
```

Negative Indices

- Python allows strings to be indexed by their position with regards to the right
  - Use negative numbers for indices.



Negative indices of the characters of the string "spam & eggs".

Negative Indices

- Example : Program illustrates negative indices.

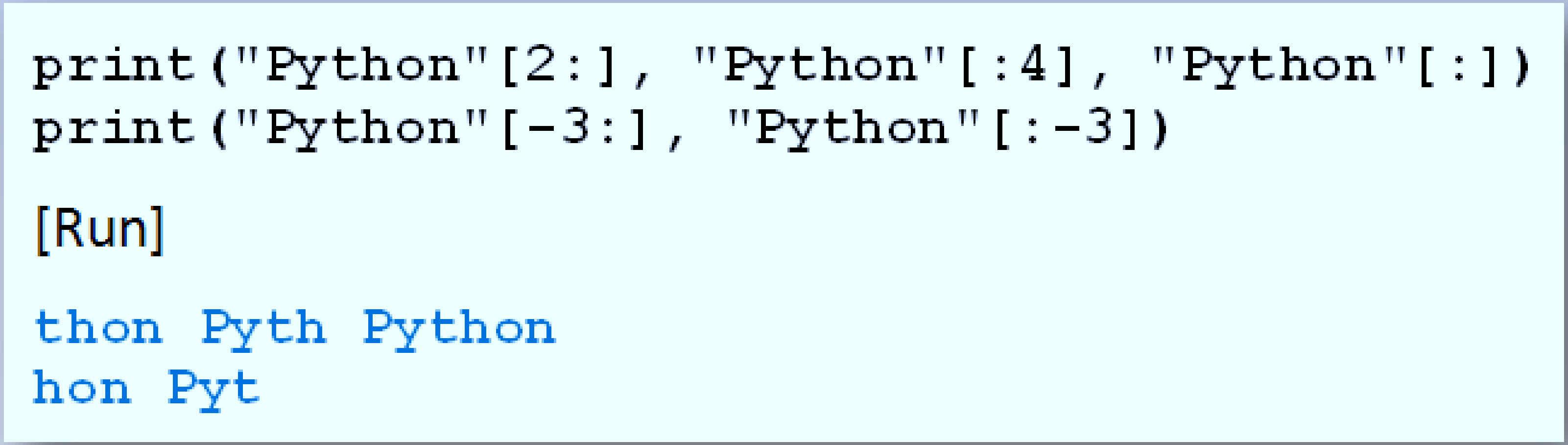
```
print("Python")
print("Python"[-1], "Python"[-4], "Python"[-5:-2])
str1 = "spam & eggs"
print(str1[-2])
print(str1[-8:-3])
print(str1[0:-1])
```

[Run]

```
Python
n t yth
g
m & e
spam & egg
```

Default Bounds for Slices

- Example 3: Program illustrates default bounds



```python
print("Python"[2:], "Python"[:4], "Python"[:])
print("Python"[-3:], "Python"[:-3])
```

[Run]

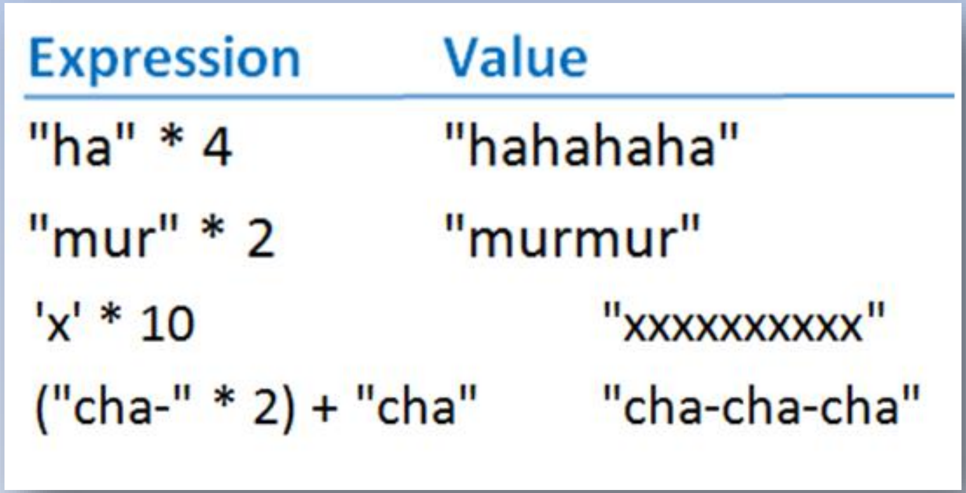
```
thon Pyth Python
hon Pyt
```

String Concatenation

- Two strings can be combined to form a new string
  - Consisting of the strings joined together
  - Represented by a plus sign
- Combination of strings, plus signs, functions, and methods can be evaluated
  - Called a string expression

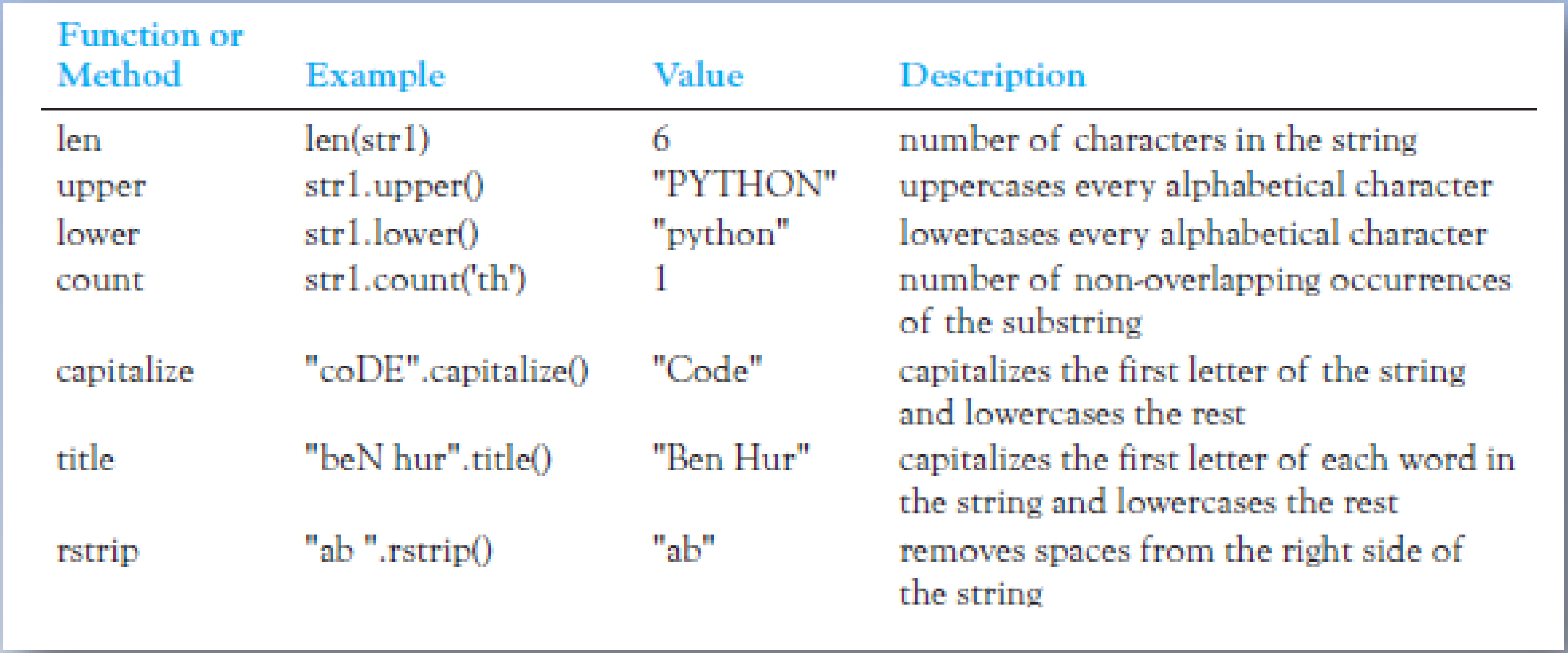
String Repetition

- Asterisk operator can be used with strings to repeatedly concatenate a string with itself



Expression	Value
"ha" * 4	"hahahaha"
"mur" * 2	"murmur"
'x' * 10	"xxxxxxxxxx"
("cha-" * 2) + "cha"	"cha-cha-cha"

String Functions and Methods



Function or Method	Example	Value	Description
len	len(str1)	6	number of characters in the string
upper	str1.upper()	"PYTHON"	uppercases every alphabetical character
lower	str1.lower()	"python"	lowercases every alphabetical character
count	str1.count('th')	1	number of non-overlapping occurrences of the substring
capitalize	"coDE".capitalize()	"Code"	capitalizes the first letter of the string and lowercases the rest
title	"beN hur".title()	"Ben Hur"	capitalizes the first letter of each word in the string and lowercases the rest
rstrip	"ab ".rstrip()	"ab"	removes spaces from the right side of the string

String Operations (str1 = "Python")

Chained Methods

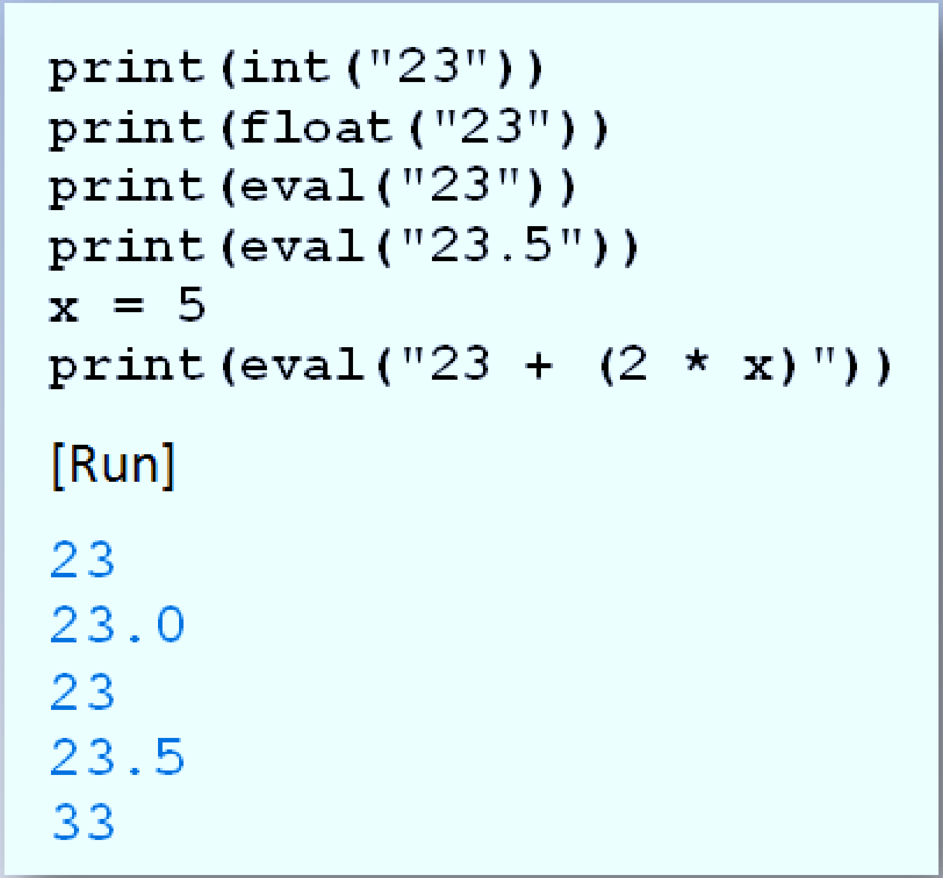
- Lines can be combined into a single line said to *chain* the two methods
  - Executed from left to right

```
praise = "Good Doggie".upper()
numberOfGees = praise.count('G')
```

```
numberOfGees = "Good Doggie".upper().count('G')
```

More String Functions

- Example : Program shows use of int, float, and eval functions

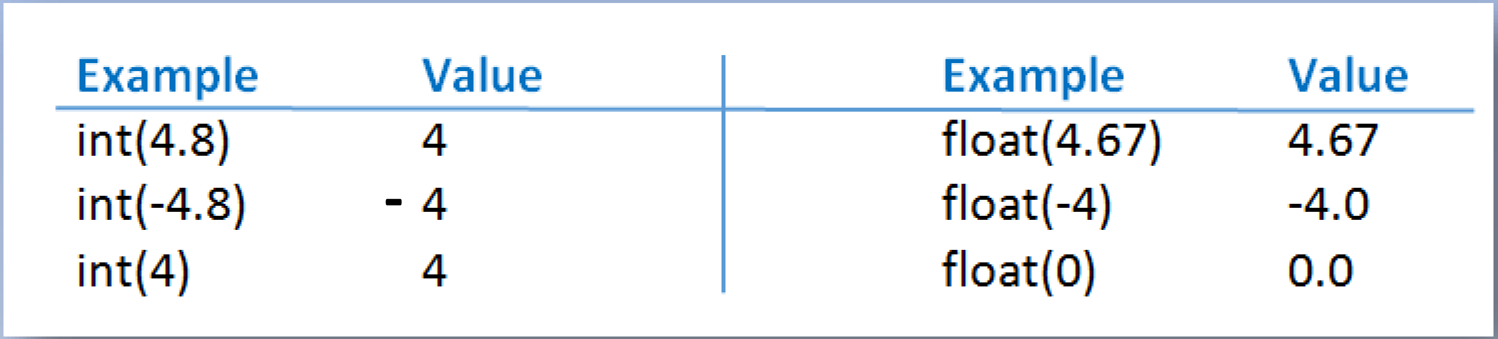


```
print(int("23"))
print(float("23"))
print(eval("23"))
print(eval("23.5"))
x = 5
print(eval("23 + (2 * x)"))
```

[Run]

```
23
23.0
23
23.5
33
```

String Functions with Numbers

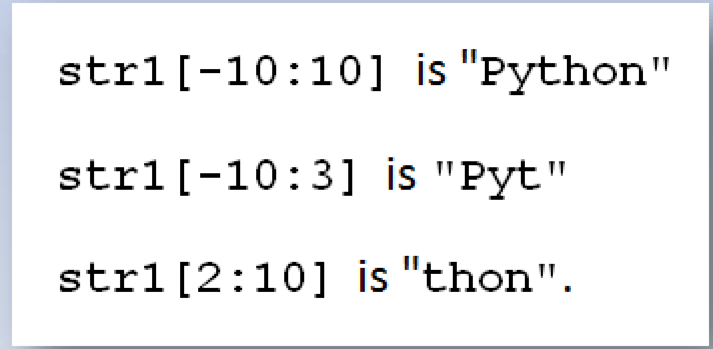


Example	Value	Example	Value
int(4.8)	4	float(4.67)	4.67
int(-4.8)	- 4	float(-4)	-4.0
int(4)	4	float(0)	0.0

- **`int`** and **`float`** also work with numbers
- The **`str`** function converts a number to its string representation

Indexing and Slicing Out of Bounds

- Python does not allow out of bounds indexing for individual characters of strings
  - Does allow out of bounds indices for slices
- Given: str1 = “Python”
  - Then `print(str1[7])` `print(str1[-7])`
  - These are OK



```
str1[-10:10] is "Python"

str1[-10:3] is "Pyt"

str1[2:10] is "thon".
```