COP 3035

Intro Programming in Python

Summer 2024

Lecture 4 – part 1

Homework 1 - Due date: 05/24/2024
Lab 2 - Due Date: 05/28/2024
Exam 1 – 05/31/2024

Lecture 4 – part 2

Review

Review

Strings

String concatenation

String multiplication

String methods (upper(), lower(), strip(), split())

Join() method

String formatting / Print formatting

String join() Method

```
separator.join(iterable)
```

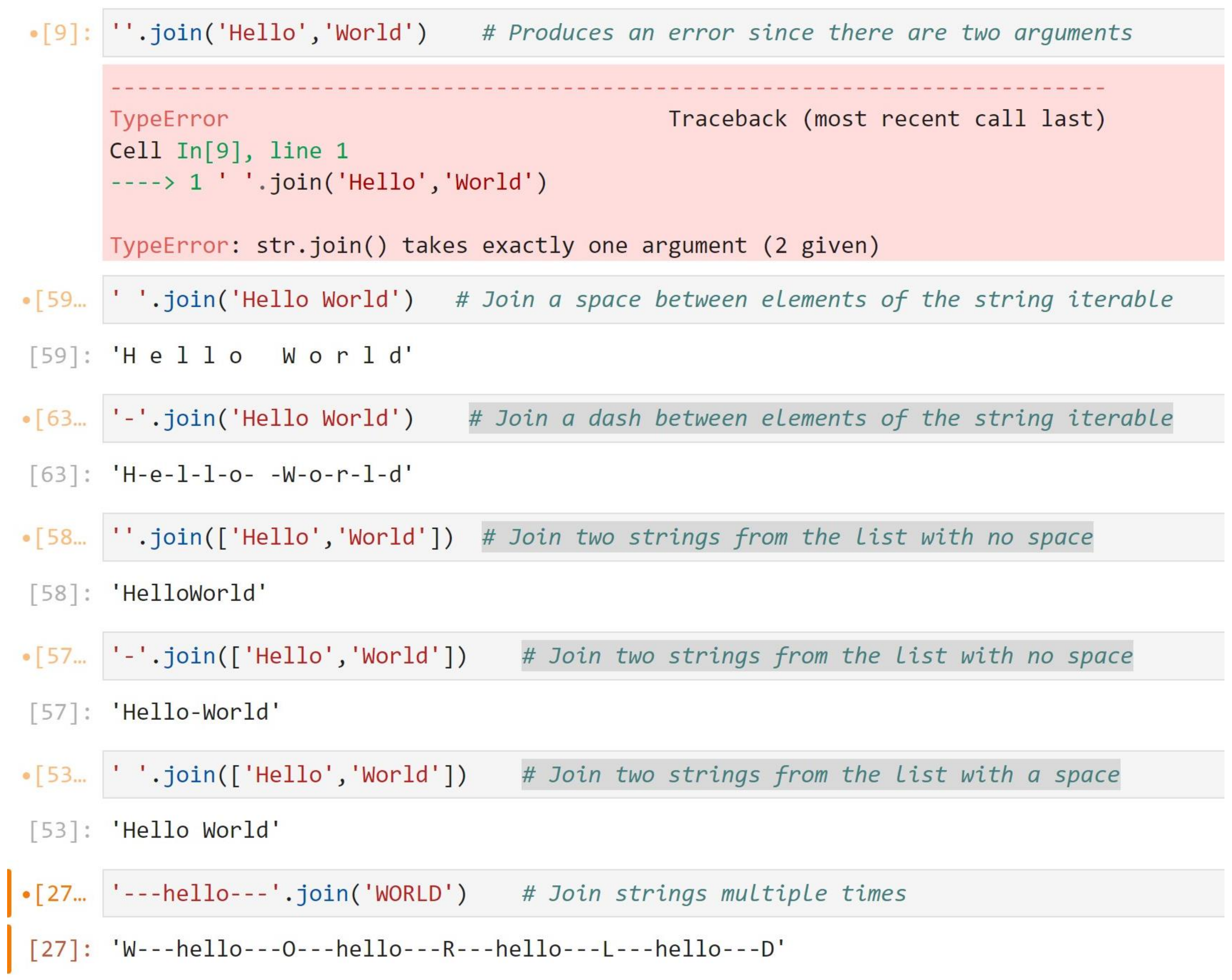
separator: A string that acts as the delimiter. It gets inserted between the elements of the iterable.

iterable: An iterable (e.g., list, tuple, set, dictionary, or even a string) containing the string elements to be joined

The output is a string where consecutive members of the iterable are joined with the separator.

Join function

separator.join(iterable)



```
•[9]: ''.join('Hello','World')    # Produces an error since there are two arguments

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[9], line 1
----> 1 ' '.join('Hello','World')

TypeError: str.join() takes exactly one argument (2 given)
```

```
•[59… ' '.join('Hello World')   # Join a space between elements of the string iterable

[59]: 'H e l l o   W o r l d'
```

```
•[63… '-'.join('Hello World')    # Join a dash between elements of the string iterable

[63]: 'H-e-l-l-o- -W-o-r-l-d'
```

```
•[58… ''.join(['Hello','World'])  # Join two strings from the list with no space

[58]: 'HelloWorld'
```

```
•[57… '-'.join(['Hello','World'])    # Join two strings from the list with no space

[57]: 'Hello-World'
```

```
•[53… ' '.join(['Hello','World'])    # Join two strings from the list with a space

[53]: 'Hello World'
```

```
•[27… '---hello---'.join('WORLD')    # Join strings multiple times

[27]: 'W---hello---O---hello---R---hello---L---hello---D'
```

Lecture 4 – part 3

Print formating

Three ways to do formatting

Formatting with placeholders

```
print('First: %s, Second: %5.2f, Third: %r' %('hi!',3.1415,'bye!'))
```

Formatting with the `.format()` method

```
print('First: {a}, Second: {b}, Third: {c}'.format(a=1,b='Two',c=12.3))
```

Formatted String Literals (f-strings)

```
print(f"My 10 character, four decimal number is:{num:{10}.{6}}")
```

Alignment, padding and precision

```
number = 40.56789

print('  {0:!^15.3f }  '.format(number))

# "{<index> : <padding character> <alignment character> <block size> <precision>}"
```

```
!!!!40.568!!!!!
```

https://docs.python.org/3/library/string.html#formatstrings

https://docs.python.org/3/reference/lexical_analysis.html#f-strings

Lecture 4 – part 3

Lists, Dictionaries, Tuples, Sets

Name	Type	Description
Integers	int	Whole numbers, such as: **3 300 200**
Floating point	float	Numbers with a decimal point: 2.3 **4.6 100.0**
Strings	str	Ordered sequence of characters: **"hello" 'Sammy' "2000" "楽しい"**
Lists	list	Ordered sequence of objects: **[10,"hello",200.3]**
Dictionaries	dict	Unordered Key:Value pairs: **{"mykey" : "value" , "name" : "Frankie"}**
Tuples	tup	Ordered immutable sequence of objects: **(10,"hello",200.3)**
Sets	set	Unordered collection of unique objects: **{"a","b"}**
Booleans	bool	Logical value indicating **True** or **False**

Lists

- Lists are ordered sequences that can hold a variety of object types.
- They are denoted by [] brackets and commas to separate objects in the list.
  [1,2,3,4,5]
- Lists support **indexing and slicing**.
- Lists can also be nested and offer a variety of useful methods that can be invoked on them.

Dictionaries

- Dictionaries are unordered mappings for storing objects.
- Dictionaries use a key-value pairing instead.
- This key-value pair allows users to quickly grab objects without needing to know an index location.
- Dictionaries use curly braces and colons to signify the keys and their associated values.

{'key1':'value1','key2':'value2'}

Tuples

- Tuples are very similar to lists.
- However, they have one key difference - **immutability**.
- Once an element is inside a tuple, it can not be reassigned.
- Tuples use parenthesis: (1,2,3)

Sets

- Sets are unordered collections of unique elements.
- Meaning there can only be one representative of the same object.

{1,2,3,4,5,’anything’}