

COP 3035

Intro Programming in Python

Summer 2024

Lecture 6 – part 1

Exam 1 – 05/31/2024 (Friday)

Lab 3 - Due Date: 06/03/2024

Homework 2 - Due date: 06/07/2024

Lecture 6 – part 2

Review

Review

Lists methods

.append()

.insert()

.remove()

.pop()

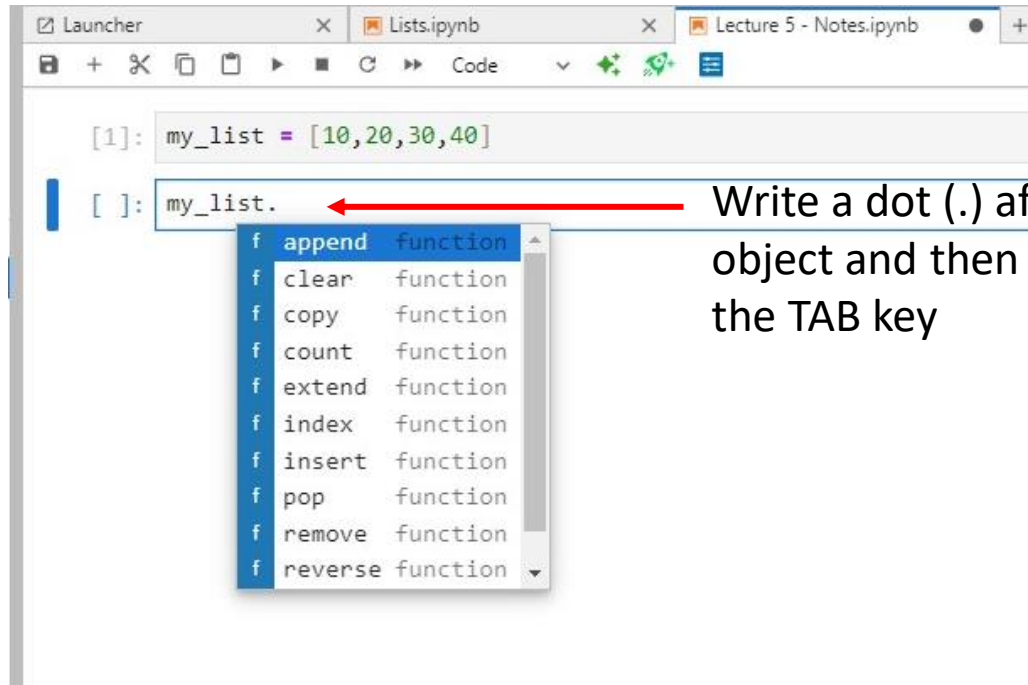
.index()

.reverse()

Join() with strings and lists

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

Accessing the object built-in functions

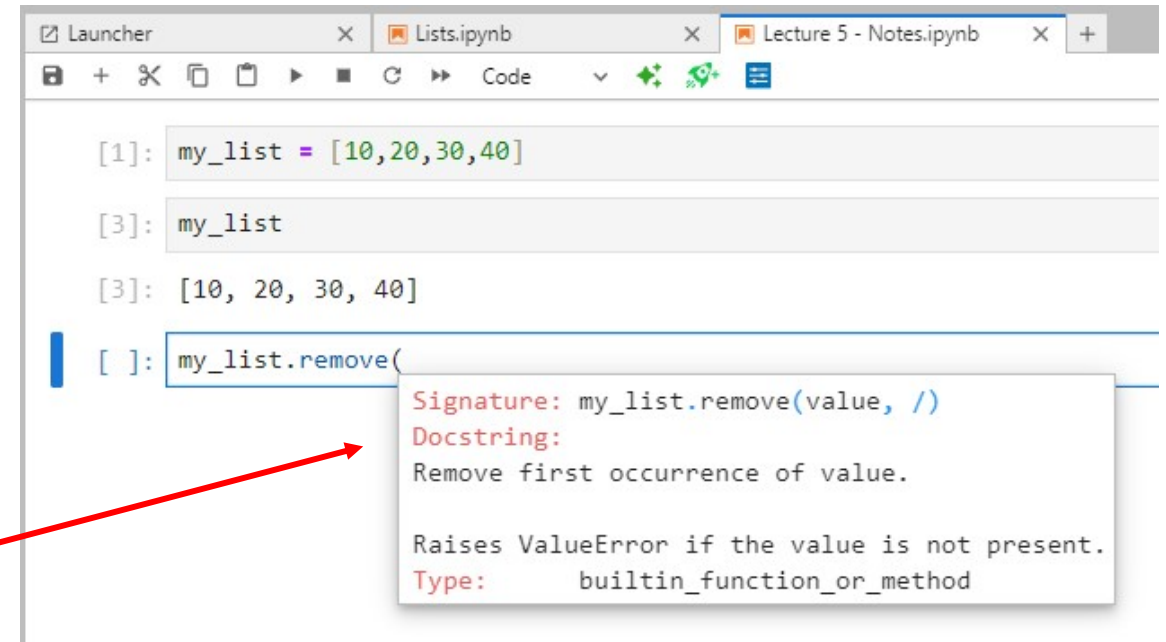


The screenshot shows a Jupyter Notebook interface with three tabs: 'Launcher', 'Lists.ipynb', and 'Lecture 5 - Notes.ipynb'. The 'Lists.ipynb' tab is active. The first code cell contains the line `[1]: my_list = [10,20,30,40]`. The second code cell contains `[]: my_list.` with a red arrow pointing to the dot. A dropdown menu is open below the code cell, listing various list methods: `append`, `clear`, `copy`, `count`, `extend`, `index`, `insert`, `pop`, `remove`, and `reverse`. Each method is preceded by a small 'f' icon and followed by the word 'function'.

Write a dot (.) after the object and then press the TAB key

Write the function until the first parenthesis and then press the Shift + TAB key

Accessing the function documentation



The screenshot shows the same Jupyter Notebook interface. The first two code cells are the same as in the previous image. The third code cell contains `[]: my_list.remove(` with a red arrow pointing to the text. A tooltip is displayed next to the code, showing the documentation for the `remove` method. The tooltip contains the following text:
Signature: `my_list.remove(value, /)`
Docstring: `Remove first occurrence of value.`
`Raises ValueError if the value is not present.`
Type: `builtin_function_or_method`

Method	Description	Example
<code>append()</code>	Adds an item to the end of the list	<code>my_list.append(4)</code>
<code>extend()</code>	Extends the list by appending elements from an iterable	<code>my_list.extend([5, 6])</code>
<code>insert()</code>	Inserts an item at a given position	<code>my_list.insert(1, 'a')</code>
<code>remove()</code>	Removes the first item with the specified value	<code>my_list.remove('a')</code>
<code>pop()</code>	Removes and returns the item at the given position	<code>item = my_list.pop(2)</code>
<code>clear()</code>	Removes all items from the list	<code>my_list.clear()</code>
<code>index()</code>	Returns the index of the first item with the specified value	<code>index = my_list.index(5)</code>
<code>count()</code>	Returns the number of items with the specified value	<code>count = my_list.count(4)</code>
<code>sort()</code>	Sorts the list in ascending order	<code>my_list.sort()</code>
<code>reverse()</code>	Reverses the elements of the list	<code>my_list.reverse()</code>
<code>copy()</code>	Returns a shallow copy of the list	<code>new_list = my_list.copy()</code>

Lecture 6 – part 3

Dictionaries

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'}`

Method	Description	Example
<code>get()</code>	Returns the value for a specified key	<code>value = my_dict.get(key)</code>
<code>update()</code>	Updates the dictionary with elements from another dictionary or iterable	<code>my_dict.update(other_dict)</code>
<code>keys()</code>	Returns a view object of the dictionary's keys	<code>keys = my_dict.keys()</code>
<code>values()</code>	Returns a view object of the dictionary's values	<code>values = my_dict.values()</code>
<code>items()</code>	Returns a view object of the dictionary's key-value pairs	<code>items = my_dict.items()</code>
<code>pop()</code>	Removes the specified key and returns its value	<code>value = my_dict.pop(key)</code>
<code>popitem()</code>	Removes and returns the last inserted key-value pair	<code>key, value = my_dict.popitem()</code>
<code>setdefault()</code>	Returns the value of a key. If the key does not exist, inserts the key with a specified value	<code>value = my_dict.setdefault(key, default_value)</code>
<code>copy()</code>	Returns a shallow copy of the dictionary	<code>new_dict = my_dict.copy()</code>
<code>clear()</code>	Removes all items from the dictionary	<code>my_dict.clear()</code>

Lecture 6 – part 4

Tuples, sets

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'}