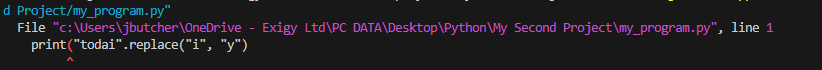
Better Error Messages

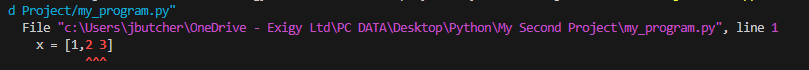
27 November 2024

18:07

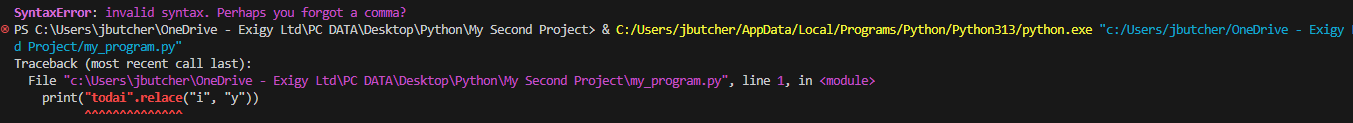
Python now shows syntax errors (e.g. unclosed bracket):



Showing missing comma in list:



And spelling mistakes:



Append Item to a List

27 November 2024

18:37

Appending is adding an item to a list, below is an example of using append:

A computer screen with text and numbers

Description automatically generated

Outcome:

A screen shot of a computer code

Description automatically generated

Index

27 November 2024

18:39

Index will search a list for a certain value. For example in the below, it will search for 8.8 in the list

\*Also note that it will search from the beginning of the list

A screen shot of a computer code

Description automatically generated

Result:

A computer screen with text

Description automatically generated

It shows the amount of items in that list with the given value.

--------------------------------------------------------------------------------------------------------------------------------------------

You can also search by steps, by using a comma in the syntax

Example:

A screen shot of a computer program

Description automatically generated

Result:

A screen shot of a computer

Description automatically generated

In this case an error is shown, as the value is skipped so no 8.8 is returned.

Accessing List Items

27 November 2024

18:49

Instead of using .\_\_getitem\_\_(1)

You can also use this syntax: list[1]

Example:

A computer screen shot of text

Description automatically generated

Result (both):

A screen shot of a computer

Description automatically generated

Accessing List Slices

27 November 2024

19:00

You can access portions of a list using a similar function to \_\_getitem\_\_ or []

Example:

A screenshot of a computer

Description automatically generated

Result:

A screen shot of a computer code

Description automatically generated

Item 0 and 5 are not shown

You can also do the below to start from 0



 Same for starting from a mid-point to the end of the list



Accessing Items and Slices with Negative Indexes

27 November 2024

19:04

Instead of counting to what intex a value is on a list (for example below 9.9 is '4'), you can use negative indexing to being from the end of the list, for example 9.9 is also '-1'

Example:

A screen shot of a computer

Description automatically generated

Result:

A screen shot of a computer code

Description automatically generated

This can also be done for slices:

A screen shot of a computer program

Description automatically generated

Result:

A screen shot of a computer code

Description automatically generated

Accessing Characters and Slices in Strings

27 November 2024

19:09

The same can be done for strings, you can also have different data types in 1 list.

Example:

A screen shot of a computer

Description automatically generated

Result:

A computer screen with text

Description automatically generated

Accessing Items in Dictionaries

27 November 2024

19:21

With dictionaries you have to slice using the key, otherwise it wont work as they don’t have indexes

Example:

A screenshot of a computer

Description automatically generated

Result:

A screen shot of a computer code

Description automatically generated

A good use case for this would be translations:

A black screen with white text

Description automatically generated

Result:

A computer screen with text on it

Description automatically generated

Converting between Data Types

27 November 2024

19:27

**Tip: Converting Between Datatypes**

Sometimes you might need to convert between different data types in Python for one reason or another. That is very easy to do:

**From tuple to list:**

* 1. >>> cool\_tuple = (1, 2, 3)
  2. >>> cool\_list = list(cool\_tuple)
  3. >>> cool\_list
  4. [1, 2, 3]

**From list to tuple:**

* 1. >>> cool\_list = [1, 2, 3]
  2. >>> cool\_tuple = tuple(cool\_list)
  3. >>> cool\_tuple
  4. (1, 2, 3)

**From string to list:**

* 1. >>> cool\_string = "Hello"
  2. >>> cool\_list = list(cool\_string)
  3. >>> cool\_list
  4. ['H', 'e', 'l', 'l', 'o']

**From list to string:**

* 1. >>> cool\_list = ['H', 'e', 'l', 'l', 'o']
  2. >>> cool\_string = str.join("", cool\_list)
  3. >>> cool\_string
  4. 'Hello'

As can be seen above, converting a list into a string is more complex. Here str() is not sufficient. We need str.join(). Try running the code above again, but this time using str.join("---", cool\_list) in the second line. You will understand how str.join() works.