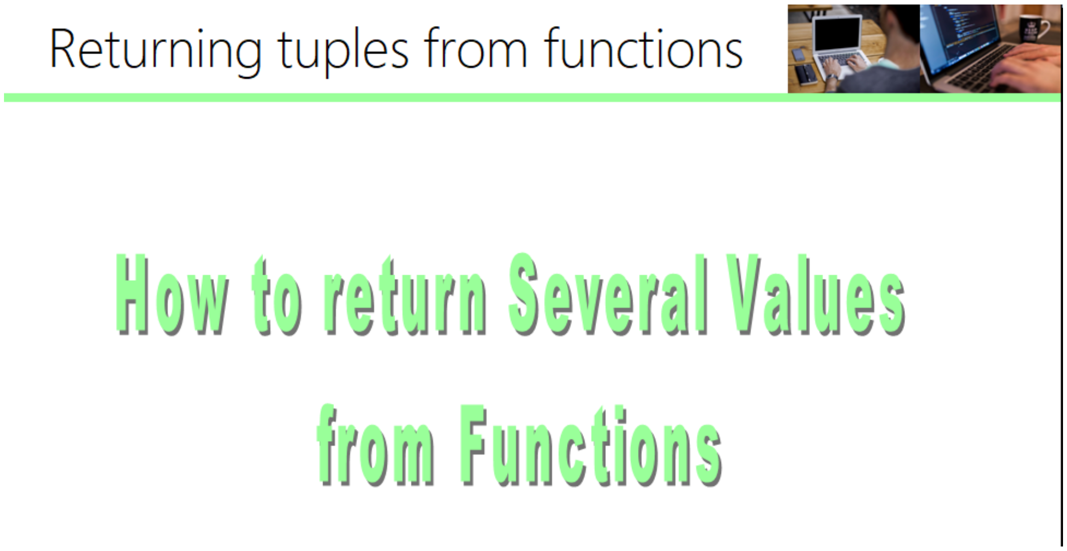
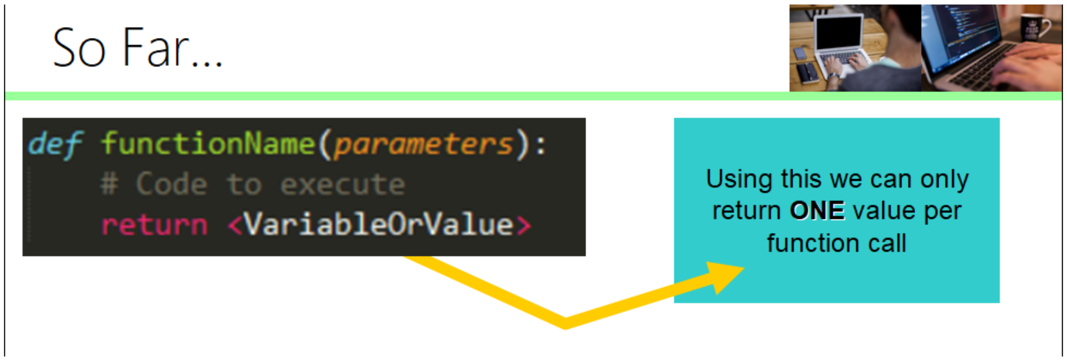


Returning tuples from functions

How to return Several Values from Functions

What we've learned so far is that we can return a value from a function to the scope that called the function and then save it to a variable for later use.

But... what if we want to return more than one value?



So Far...

```
def functionName(parameters):
    # Code to execute
    return <VariableOrValue>
```

Using this we can only return **ONE** value per function call

This is where our newly discovered friends come into play, **TUPLES AND LISTS!**

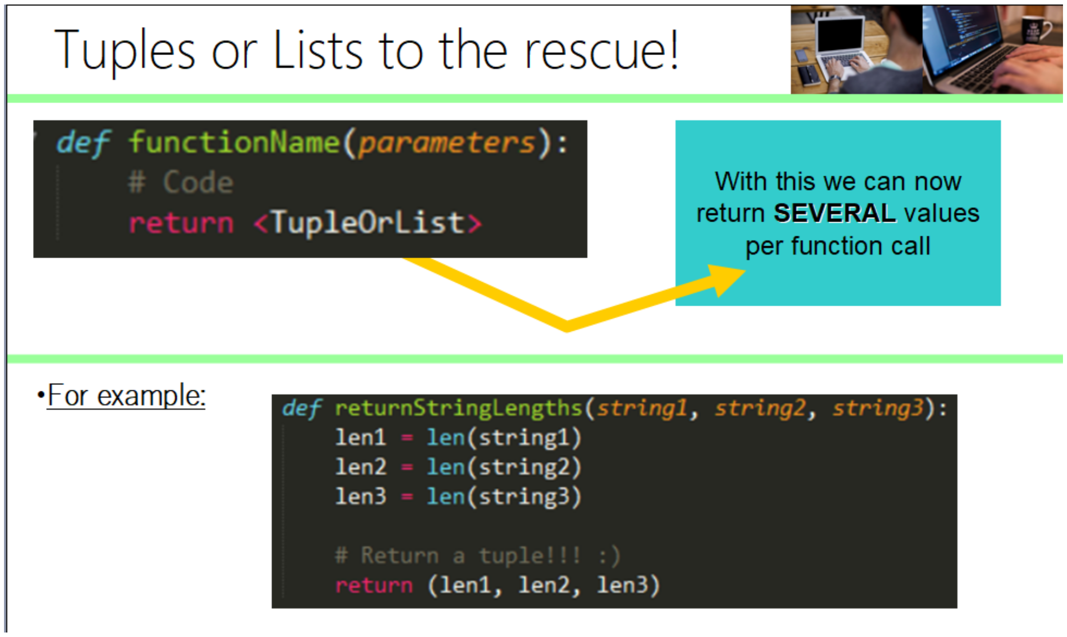
If we return a tuple or a list from a function, we can return several values and assign them individually to their corresponding variables

(We will see this in detail in the next diagrams).

We've learned that we can display the values returned by a function using print().

Since lists and tuples are iterables, we can iterate over their elements using a for/in loop like the examples below. We can also access each individual element by its corresponding index.

USE VARIABLES TO STORE VALUES RETURNED



Tuples or Lists to the rescue!

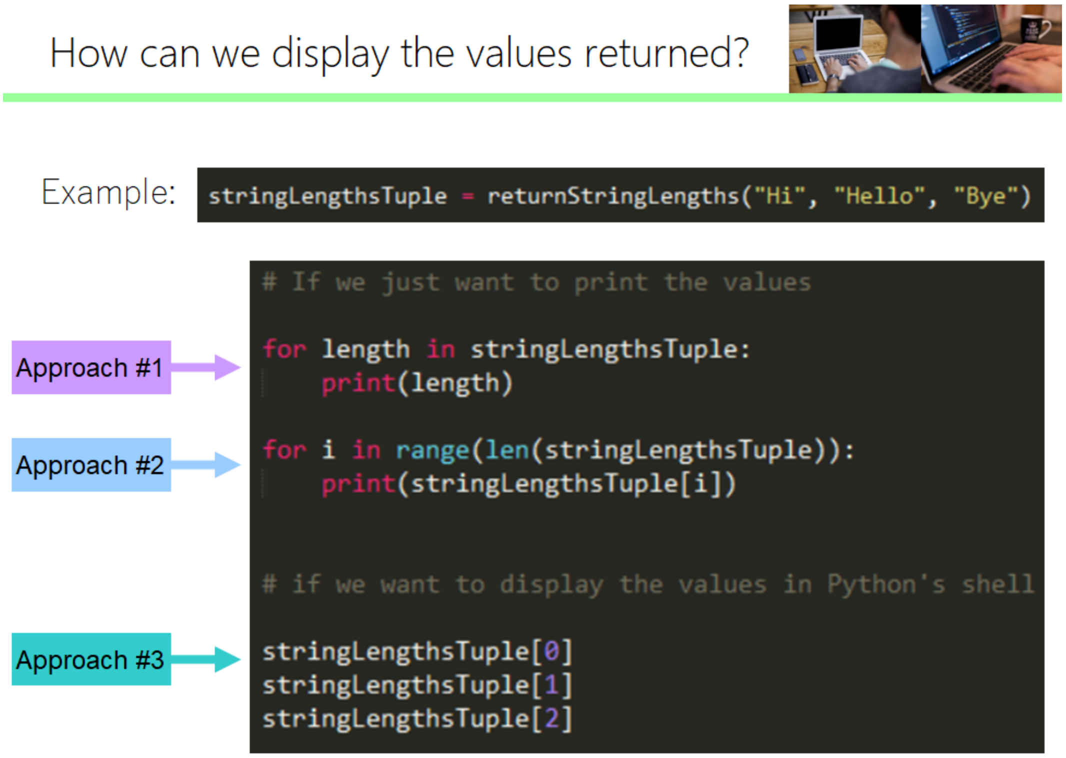
```
def functionName(parameters):
    # Code
    return <TupleOrList>
```

With this we can now return **SEVERAL** values per function call

•For example:

```
def returnStringLengths(string1, string2, string3):
    len1 = len(string1)
    len2 = len(string2)
    len3 = len(string3)

    # Return a tuple!!! :)
    return (len1, len2, len3)
```



How can we display the values returned?

Example:

```
stringLengthsTuple = returnStringLengths("Hi", "Hello", "Bye")
```

Approach #1

Approach #2

Approach #3

```
# If we just want to print the values

for length in stringLengthsTuple:
    print(length)

for i in range(len(stringLengthsTuple)):
    print(stringLengthsTuple[i])


# if we want to display the values in Python's shell

stringLengthsTuple[0]
stringLengthsTuple[1]
stringLengthsTuple[2]
```

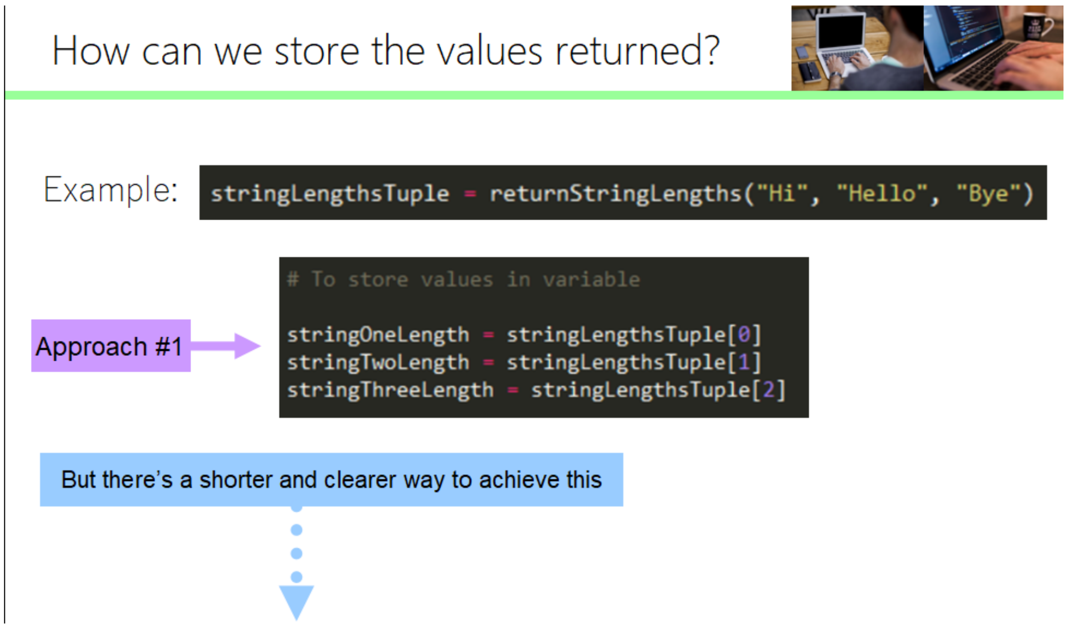
But... wait a minute! What if we want to store the values returned by the function call? Can we apply the principles we've learned? Yes, you can. But there is a new syntax that will allow you to store each element in the tuple into its corresponding variable.

Let's see why we need this...

In this example, we are using the function defined above, that returns a tuple with the length of each argument. In this case, (2, 5, 3).

To store them in separate variables, we would first need to create a new variable to store the entire tuple and then index individually and assign them to their variable. This is one way to achieve this but...

Keep reading to find out how you can do this in ONE LINE!



How can we store the values returned?

Example:

```
stringLengthsTuple = returnStringLengths("Hi", "Hello", "Bye")
```

Approach #1

```
# To store values in variable

stringOneLength = stringLengthsTuple[0]
stringTwoLength = stringLengthsTuple[1]
stringThreeLength = stringLengthsTuple[2]
```

But there's a shorter and clearer way to achieve this

To assign the variables individually in ONE LINE:

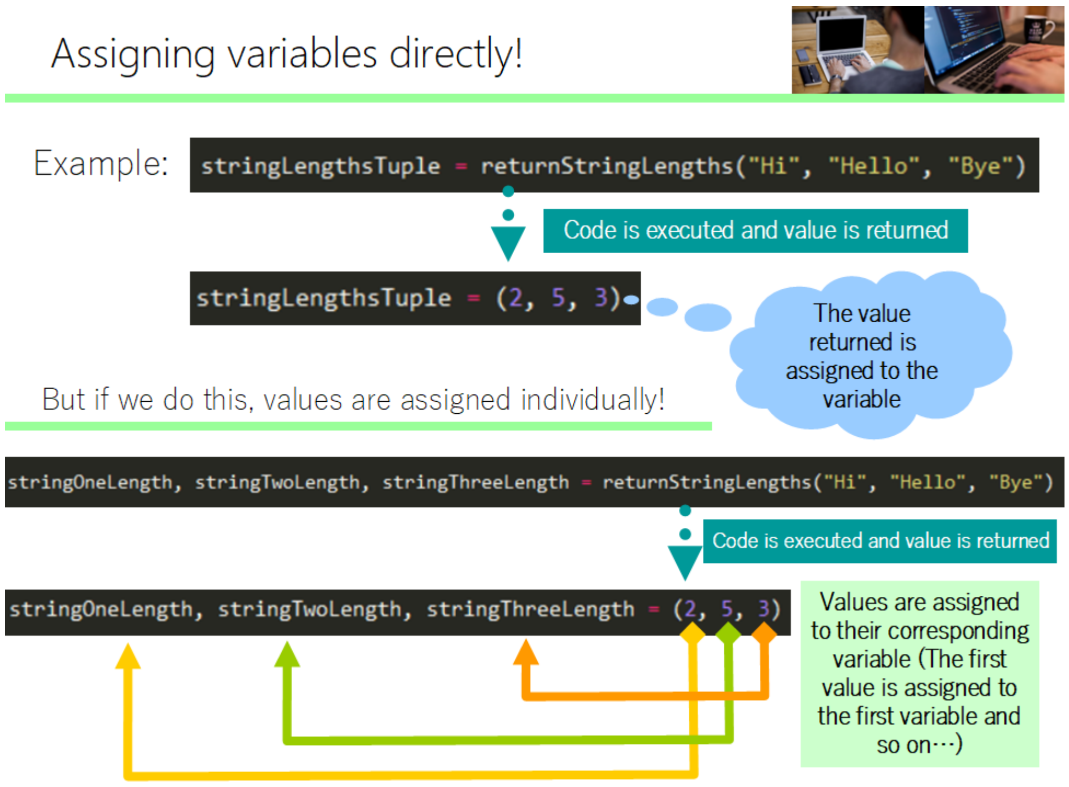
- On the left hand side we create as many variables as the number of values contained in the tuple returned by the function call. (In this case, 3 variables for a tuple that contains 3 values)
- On the right hand side we call the function

On the diagram below you can see this process broken down into steps with a "Behind the scenes" look of what happens when the value is returned.

Let's check these values in Python's shell. As you can see, the values are assigned individually and you can now use these variables later in your code. Yes! : D

TIPS

You can convert the arguments you pass into a function into a tuple by adding an asterisk before the function's parameter. This will convert all the arguments you pass in separated by commas into a single tuple that you can use inside your



Assigning variables directly!

Example:

```
stringLengthsTuple = returnStringLengths("Hi", "Hello", "Bye")
```

Code is executed and value is returned

```
stringLengthsTuple = (2, 5, 3)
```

The value returned is assigned to the variable

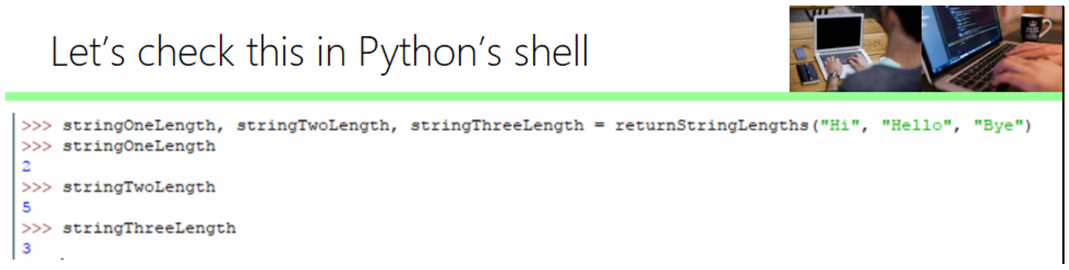
But if we do this, values are assigned individually!

```
stringOneLength, stringTwoLength, stringThreeLength = returnStringLengths("Hi", "Hello", "Bye")
```

Code is executed and value is returned

```
stringOneLength, stringTwoLength, stringThreeLength = (2, 5, 3)
```

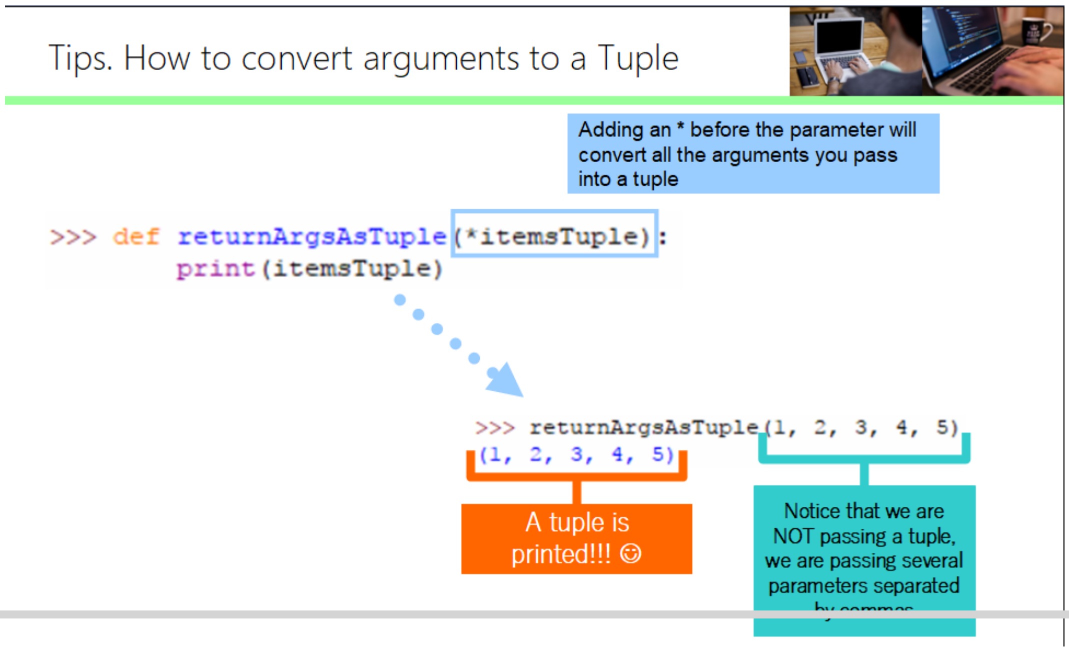
Values are assigned to their corresponding variable (The first value is assigned to the first variable and so on…)



Let's check this in Python's shell

```
>>> stringOneLength, stringTwoLength, stringThreeLength = returnStringLengths("Hi", "Hello", "Bye")
>>> stringOneLength
2
>>> stringTwoLength
5
>>> stringThreeLength
3
```

function.



Tips. How to convert arguments to a Tuple

Adding an * before the parameter will convert all the arguments you pass into a tuple

```
>>> def returnArgsAsTuple(*itemsTuple):
        print(itemsTuple)
```

```
>>> returnArgsAsTuple(1, 2, 3, 4, 5)
(1, 2, 3, 4, 5)
```

A tuple is printed!!! ☺

Notice that we are NOT passing a tuple, we are passing several parameters separated by commas

Hope this helps!

If you have any questions, please do not hesitate to post them in forums or right below this post. Community TAs and your fellow classmates will be there to help you :)

Estefania.

This post is visible to everyone.

Add a Response

6 responses

LuisGo17
4 months ago

+
...

Thanks for your time and effort trying to make this course easier!

Thank you so much for your kind words Luis, I really appreciate it 😊 I'm very glad it has been helpful.

...

Estefania.

posted 4 months ago by **Kiara-Elizabeth** (Community TA)

Add a comment

Keiichiart
4 months ago

+
...

Brilliant!! Thanks!

Thank you very much Keiichiart, I'm very glad it helped 🙂

...

Estefania.

posted 4 months ago by **Kiara-Elizabeth** (Community TA)

Add a comment

lyking
4 months ago

Clearly and funny! Thanks for your sharing

Thank you very much for your comment 🙂 , I'm very glad it helped.

Estefania.

posted 4 months ago by **Kiara-Elizabeth** (Community TA)

Great article. It made tuples a lot clearer !!

posted 4 months ago by **tech1trans**

Add a comment

tech1trans
4 months ago

Great article. It made tuples a lot clearer !!

Thank you very much, I'm very glad it helped 🙂

Estefania.

posted 4 months ago by **Kiara-Elizabeth** (Community TA)

Add a comment

colourlesslight
3 months ago

Estefania, your guides are very clear and a big help. Thank you :D

p.s. I enjoy all the smiley faces in your tutorials.

Thank you very much for your very kind words 😊 I'm very glad they been helpful.

Estefania.

posted 3 months ago by **Kiara-Elizabeth** (Community TA)

Add a comment

kitseason
3 months ago

Thanks to this, I practiced with this silly code, which helped cement the concepts:

```
def testTuple(red, yellow):
    """
        take in two parameters-- red, yellow -- of any type
        and return a tuple to assign to two variables
    """
    blue = red
    purple = yellow
    print(blue + str(purple))
    color = ()

    color = color + (blue,)
    print(color)
    color = color + (purple,)
    print(color)

    return (color)

pink = 'black'
sienna = 3
orange, green = testTuple(pink, sienna)

print(orange)
print(green)
```

I'm very glad it helped 🙂 It's great that you are experimenting with these examples. I love practicing with short examples because really help us deepen our understanding and discover amazing new things that we didn't even know were possible. 👍

Estefania.

posted 3 months ago by **Kiara-Elizabeth** (Community TA)

Add a comment

Showing all responses

Add a response:

Preview

Submit