• percipio51_The map() Function.py - Python - Visual Studio Code

File Edit Selection View Go Debug Tasks Help

EXPLORER

OPEN EDITORS 1 UNSAVED

Welcome

percipio51_The map() Fun..

PYTHON

Automate-Boring-Stuff

my_code

Percipio_Python3-Course

01_Start

02_Data-Sequence Types

03_Collections-Mapping-Lo..

04_Modules-Functions

05_Classes

06_Working-with-Files

07_Comprehensions

08_Iterables-and-Generators

percipio50_Basic Iteration..

percipio51_The map() Fu...

percipio52_The Filter() Fu...

percipio53_The functools...

percipio54_Implementing..

percipio55_Implement an..

percipio56_Implement an..

percipio57_Simple Gener...

percipio58_Lazy Generat...

percipio59_Recursive Gen..

percipio60_Exercise-Crea...

09_Exceptions

10_Automation Programmi...

Python Projects_2014

CMD_Python_Set-Path.txt

Python_Clear-Window-Comm..

python_exercises_00.py

python_exercises_01.py

Python_Tutorial_Running-Scri...

Python_Tutorials.md

start_code_for_python.py

Welcome | percipio51_The map() Function.py

```python
'''
percipio51_.py
Percipio video: Iterables-and-Generators;

* Demonstrating the map() function
* The map() function applies the function to every element of one or more sequences
* The map() function is an iterator which allows iteration over each element in a list
* Code to capitalize the first letter of each word
* Lambda functions are anonymous functions that can be defined in line.  They avoid having to define a function which will only be used once.
    * Lambda functions are limited to a single expression for the value they can return
    * Original functions are more powerful because they can be defined in multiple lines with sophisticated flow control logic
'''
nl = '\n'
vehicles = ['sedan', 'coupe', 'hatchback', 'wagon'] # starting sequence
print('vehicles list_starting sequence:', vehicles)
cars = map(str.capitalize, vehicles) # capitalizes the beginning letter of each element in the vehicles list. Creates a cars map by mapping the string function 'capitalize' to every element in vehicles
print('Map object "cars":', cars)
# using a function built into the string class
cars = list(map(str.capitalize, vehicles)) # apply the list() function to the map() function as it applies string-capitalize to every vehicle in the list
print('List object cars, with the first letter of each element capitalized:', cars)

# Generating the same 'cars' list, instead of using the string-function->capitalize, use basic list comprehensions to write the previous two lins of code.
# using an instance if a string, a method
cars = [car.capitalize() for car in vehicles] # The instance variable (car-#1) for the instance (car-#2) in the list (vehicles), and than call a funtion or method on it, (capitalize() method).  The parentheses calls that method FOR each CAR IN the VEHICLES list.
print('List object cars using comprehension:', cars)

print(nl, 'Using original functions with the map() function')
def quad(val):
```

• percipio51_The map() Function.py - Python - Visual Studio Code

File Edit Selection View Go Debug Tasks Help

EXPLORER

OPEN EDITORS 1 UNSAVED

Welcome

percipio51_The map() Fun..

PYTHON

Automate-Boring-Stuff

my_code

Percipio_Python3-Course

01_Start

02_Data-Sequence Types

03_Collections-Mapping-Lo..

04_Modules-Functions

05_Classes

06_Working-with-Files

07_Comprehensions

08_Iterables-and-Generators

percipio50_Basic Iteration..

percipio51_The map() Fu...

percipio52_The Filter() Fu...

percipio53_The functools...

percipio54_Implementing..

percipio55_Implement an..

percipio56_Implement an..

percipio57_Simple Gener...

percipio58_Lazy Generat...

percipio59_Recursive Gen..

percipio60_Exercise-Crea...

09_Exceptions

10_Automation Programmi...

Python Projects_2014

CMD_Python_Set-Path.txt

Python_Clear-Window-Comm..

python_exercises_00.py

python_exercises_01.py

Python_Tutorial_Running-Scri...

Python_Tutorials.md

start_code_for_python.py

Welcome | percipio51_The map() Function.py

```python
def quad(val):
    return val ** 4 # original function created which receives a value and returns that value modified # '** 4' is a square of a square

nums = range(4) # variable created with 4 numbers (0,1,2,3)
print('List of nums:', list(nums)) # unmoodified number list
quads = list(map(quad, nums)) # Using the map() function, generate a 'quads' list by applying the list() function to map, where map() is mapping the quad function to that sequence of numbers.
''' # ?? WHERE IS QUAD (WITHOUT s) FROM AND HOW DOES THE PREVIOUS LINE KNOW WHAT QUAD IS WHEN QUADS IS IDENTIFIED ?? '''
print('List of quads:', quads) # each number is a square of a square of the original number
# lambda functions
quads = list(map(lambda val: val ** 4, nums)) # generate a list of quads by applying the list() function to the map objects that gets created.  The map () function calls the lambda function where val is similar to the val from the original function above.  Here, val is the parameter that will be received from the one sequence, nums.  The colon seperates the lambda parameter from the lambda expression.
print('List of quads generated with a lambda function:', nl, quads) #

# Function with 2 parameters and want map() to apply them
print(nl, 'Functions with 2 parameters')
def mult(x, y): # function requiring 2 parameters
    return x * y # returns nums * quads
num_quads = list(map(mult, nums, quads)) # use map() to apply a function (mult), without parentheses, to not one sequence, but two (i.e.: x as 'nums' and y as 'quads', etc.)
print('List of num_quads:', num_quads) # list of num_quads that is a number mutlipied by the quad (i.e.: 0x0=0, 1x1=1, 2x16=32, 3x81=243)
# previous code done in a lambda function
num_quads = list(map(lambda x, y: x * y, nums, quads)) # using two parameters, two parameters must be named (i.e.: x,y), after ambda, commaseparated.  The colon separates the parameters for the lambda function from the expression that will return a value. In the end code section (x * y, nums, quads) the x comes from the nums this and the y comes from the quads list.
print('List of num_quads generated with a lambda function:', nl, num_quads) # same list should be printed
'''
```

• percipio51_The map() Function.py - Python - Visual Studio Code

File Edit Selection View Go Debug Tasks Help

EXPLORER

OPEN EDITORS 1 UNSAVED

Welcome

• percipio51_The map() Fun..

PYTHON

Automate-Boring-Stuff

my_code

Percipio_Python3-Course

01_Start

02_Data-Sequence Types

03_Collections-Mapping-Lo..

04_Modules-Functions

05_Classes

06_Working-with-Files

07_Comprehensions

08_Iterables-and-Generators

percipio50_Basic Iteration..

percipio51_The map() Fu...

percipio52_The Filter() Fu...

percipio53_The functools...

Welcome | percipio51_The map() Function.py •

```
'''
vehicles list_starting sequence: ['sedan', 'coupe', 'hatchback', 'wagon']
Map object "cars": <map object at 0x00000209690C8C50>
List object cars, with the first letter of each element capitalized: ['Sedan', 'Coupe',
    'Hatchback', 'Wagon']
List object cars using comprehension: ['Sedan', 'Coupe', 'Hatchback', 'Wagon']

 Using original functions with the map() function
List of nums: [0, 1, 2, 3]
List of quads: [0, 1, 16, 81]
List of quads generated with a lambda function:
 [0, 1, 16, 81]

 Functions with 2 parameters
List of num_quads: [0, 1, 32, 243]
List of num_quads generated with a lambda function:
 [0, 1, 32, 243]
'''
```