course_3_assessment_2

Due: 2018-11-25 01:36:00

Description: Assessment for the More On Accumulation lesson.

Score: 0 of 7 = 0.0%

Questions

Not yet graded

Write code to assign to the variable `map_testing` all the elements in lst_check while adding the string “Fruit: ” to the beginning of each element using mapping.

Save & Run   11/10/2020, 12:55:38 AM - 3 of 3   Show in CodeLens

```

lst_check = ['plums', 'watermelon', 'kiwi', 'strawberries', 'blueberries', 'peaches', 'app

map_testing = list(map(lambda x: 'Fruit: ' + x, lst_check))
```

ActiveCode (ac21_7_1)

Result	Actual Value	Expected Value	Notes	
Pass	['Fru...aya']	['Fru...aya']	Testing that map_testing has the correct values.	Expand Differences
Pass	'map('	'\nlst_...eck))'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'filter('	'\nlst_...eck))'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'sum('	'\nlst_...eck))'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'zip('	'\nlst_...eck))'	Testing your code (Don't worry about actual and expected values).	Expand Differences

You passed: 100.0% of the tests

Not yet graded

Below, we have provided a list of strings called `countries`. Use filter to produce a list called `b_countries` that only contains the strings from `countries` that begin with B.

Save & Run   11/10/2020, 12:56:38 AM - 2 of 2   Show in CodeLens

```

countries = ['Canada', 'Mexico', 'Brazil', 'Chile', 'Denmark', 'Botswana', 'Spain', 'Brita
b_countries = list(filter(lambda x: x[0] == 'B', countries))

```

ActiveCode (ac21_7_2)

Result	Actual Value	Expected Value	Notes	
Pass	['Bra...ium']	['Bra...ium']	Testing that b_countries is correct.	Expand Differences
Pass	'map('	'\ncoun...es))\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'filter('	'\ncoun...es))\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'sum('	'\ncoun...es))\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'zip('	'\ncoun...es))\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences

You passed: 100.0% of the tests

Not yet graded

Below, we have provided a list of tuples that contain the names of Game of Thrones characters. Using list comprehension, create a list of strings called `first_names` that contains only the first names of everyone in the original list.

Save & Run   11/10/2020, 12:57:35 AM - 3 of 3   Show in CodeLens

```

people = [('Snow', 'Jon'), ('Lannister', 'Cersei'), ('Stark', 'Arya'), ('Stark', 'Robb'),
first_names = [x[1] for x in people]

```

ActiveCode (ac21_7_3)

Result	Actual Value	Expected Value	Notes	
Pass	['Jon...ter']	['Jon...ter']	Testing that first_names is correct.	Expand Differences
Pass	'map('	'\npeop...ple]\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'filter('	'\npeop...ple]\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'sum('	'\npeop...ple]\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'zip('	'\npeop...ple]\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences

You passed: 100.0% of the tests

Not yet graded

Use list comprehension to create a list called `lst2` that doubles each element in the list, `lst`.

Save & Run 11/10/2020, 12:58:10 AM - 2 of 2 Show in CodeLens

```

lst = [["hi", "bye"], "hello", "goodbye", [9, 2], 4]
lst2 = [x*2 for x in lst]

```

ActiveCode (ac21_7_4)

Result	Actual Value	Expected Value	Notes	
Pass	[['hi...], 8]	[['hi...], 8]	Testing that lst2 is assigned to correct values	Expand Differences
Pass	'map('	'\nlst ...lst]\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'filter('	'\nlst ...lst]\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'sum('	'\nlst ...lst]\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'zip('	'\nlst ...lst]\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences

You passed: 100.0% of the tests

Not yet graded

Below, we have provided a list of tuples that contain students' names and their final grades in PYTHON 101. Using list comprehension, create a new list `passed` that contains the names of students who passed the class (had a final grade of 70 or greater).

Save & Run 11/10/2020, 12:59:01 AM - 2 of 2 Show in CodeLens

```
students = [('Tommy', 95), ('Linda', 63), ('Carl', 70), ('Bob', 100), ('Raymond', 50), ('S
passed = [x[0] for x in students if x[1] >= 70]
```

ActiveCode (ac21_7_5)

Result	Actual Value	Expected Value	Notes	
Pass	['Tom...Sue']	['Tom...Sue']	Testing that passed is correct.	Expand Differences
Pass	'map('	'\nstud... 70]\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'filter('	'\nstud... 70]\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'sum('	'\nstud... 70]\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'zip('	'\nstud... 70]\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences

You passed: 100.0% of the tests

Not yet graded

Write code using zip and filter so that these lists (l1 and l2) are combined into one big list and assigned to the variable `opposites` if they are both longer than 3 characters each.

Save & Run 11/10/2020, 1:02:14 AM - 4 of 4 Show in CodeLens

```
l1 = ['left', 'up', 'front']
l2 = ['right', 'down', 'back']
opposites = list(filter(lambda x: (len(x[0]) > 3 and len(x[1]) > 3), zip(l1, l2)))
```

ActiveCode (ac21_7_6)

Result	Actual Value	Expected Value	Notes	
Pass	[('le...ck')]	[('le...ck')]	Testing that opposites has the correct list of tuples.	Expand Differences
Pass	'map('	'\nl1 =...2)))\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'filter('	'\nl1 =...2)))\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'sum('	'\nl1 =...2)))\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'zip('	'\nl1 =...2)))\n'	Testing your code (Don't worry about actual and expected values).	Expand Differences

You passed: 100.0% of the tests

Not yet graded

Below, we have provided a `species` list and a `population` list. Use zip to combine these lists into one list of tuples called `pop_info`. From this list, create a new list called `endangered` that contains the names of species whose populations are below 2500.

Save & Run   11/10/2020, 1:05:03 AM - 3 of 3   Show in CodeLens

```

species = ['golden retriever', 'white tailed deer', 'black rhino', 'brown squirrel', 'fiel

population = [10000, 90000, 1000, 2000000, 500000, 500, 1200, 8000, 12000, 2300, 7500, 100

pop_info = list(zip(species, population))
endangered = [x[0] for x in pop_info if x[1] < 2500]
```

ActiveCode (ac21_7_7)

Result	Actual Value	Expected Value	Notes	
Pass	[('go...000)]	[('go...000)]	Testing that pop_info was created correctly.	Expand Differences
Pass	'map('	'\nspec...2500]'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'filter('	'\nspec...2500]'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'sum('	'\nspec...2500]'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'zip('	'\nspec...2500]'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	['bla...tle']	['bla...tle']	Testing that endangered was created correctly.	Expand Differences

You passed: 100.0% of the tests

Score Me

username: sandipan.dey@gmail.com | Back to top