

## Filter() and Lambda Functions

Learn / Courses / Introduction to Functions in Python

← Course Outline →

Daily XP 1950

Exercise

### Map() and lambda functions

So far, you've used lambda functions to write short, simple functions as well as to redefine functions with simple functionality. The best use case for lambda functions, however, are for when you want these simple functionalities to be anonymously embedded within larger expressions. What that means is that the functionality is not stored in the environment, unlike a function defined with `def`. To understand this idea better, you will use a lambda function in the context of the `map()` function.

Recall from the video that `map()` applies a function over an object, such as a list. Here, you can use lambda functions to define the function that `map()` will use to process the object. For example:

```
nums = [2, 4, 6, 8, 10]
result = map(lambda a: a ** 2, nums)
```

You can see here that a lambda function, which raises a value `a` to the power of 2, is passed to `map()` alongside a list of numbers, `nums`. The map object that results from the call to `map()` is stored in `result`. You will now practice the use of lambda functions with `map()`. For this exercise, you will map the functionality of the `add_bangs()` function you defined in previous exercises over a list of strings.

Instructions 100 XP

- In the `map()` call, pass a lambda function that concatenates the string `!!!` to a string `item`; also pass the list of strings, `spells`. Assign the resulting map object to `shout_spells`.
- Convert `shout_spells` to a list and print out the list.

Take Hint (-30 XP)

script.py

Light Mode

```
1 # Create a list of strings: spells
2 spells = ["protego", "accio", "expecto patronum", "legilimens"]
3
4 # Use map() to apply a lambda function over spells: shout_spells
5 ____ = map(____, ____ )
6
7 # Convert shout_spells to a list: shout_spells_list
8 ____
9
10 # Print the result
11 print(shout_spells_list)
```

↻ Run Code Submit Answer

IPython Shell

Slides

In [1]:

### Question:

In the previous exercise, you used lambda functions to anonymously embed an operation within `map()`.

You will practice this again in this exercise by using a lambda function with `filter()`, which may be new to you!

The function `filter()` offers a way to filter out elements from a list that don't satisfy certain criteria.

Your goal in this exercise is to use `filter()` to create, from an input list of strings, a new list that contains only strings that have more than 6 characters.

**\*\*Instructions:\*\***

1. In the `filter()` call, pass a lambda function and the list of strings, `fellowship`.  
The lambda function should check if the number of characters in a string member is greater than 6 (use the `len()` function to do this).  
Assign the resulting filter object to `result`.
2. Convert `result` to a list and print out the list.

## Answer:

```
# Create a list of strings: fellowship
fellowship = ['frodo', 'samwise', 'merry', 'pippin', 'aragorn', 'boromir',
              'legolas', 'gimli', 'gandalf']

# Use filter() to apply a lambda function over fellowship: result
result = filter(lambda member: len(member) > 6, fellowship)

# Convert result to a list: result_list
result_list = list(result)

# Print result_list
print(result_list)
```

## Explanation:

1. **List of Strings:**  
The input list `fellowship` contains names of characters from "The Lord of the Rings".  
Example: `['frodo', 'samwise', 'merry', 'pippin', 'aragorn', 'boromir', 'legolas', 'gimli', 'gandalf']`.
2. **Lambda Function:**  
A lambda function is used to check whether the length of each string is greater than 6.  
Syntax: `lambda member: len(member) > 6`.
3. **filter():**  
The `filter()` function applies the lambda function to each element of the list.  
Only elements for which the lambda function returns `True` are included in the result.  
For example, `"samwise"` and `"aragorn"` satisfy the condition `len(member) > 6`, so they are included.

4. **\*\*Convert to List:\*\***

The ``filter()`` function returns a filter object, which is then converted to a list using ``list()``.

Example Output: ``['samwise', 'aragorn', 'boromir', 'legolas', 'gandalf']``.

5. **\*\*Print Result:\*\***

The resulting list is printed to verify the output.

The expected output is: ``['samwise', 'aragorn', 'boromir', 'legolas', 'gandalf']``.