23.3. Filter

Now consider another common pattern: going through a list and keeping only those items that meet certain criteria. This is called a filter.

Save & Run Original - 1 of 1 Show in CodeLens

```
def keep_evens(nums):
    new_list = []
    for num in nums:
        if num % 2 == 0:
            new_list.append(num)
    return new_list

print(keep_evens([3, 4, 6, 7, 0, 1]))
```

```
[4, 6, 0]
```

Activity: 1 -- ActiveCode (ac21_3_1)

Again, this pattern of computation is so common that Python offers a more compact and general way to do it, the `filter` function. `filter` takes two arguments, a function and a sequence. The function takes one item and return True if the item should. It is automatically called for each item in the sequence. You don't have to initialize an accumulator or iterate with a for loop.

Save & Run Original - 1 of 1 Show in CodeLens

```
def keep_evens(nums):
    new_seq = filter(lambda num: num % 2 == 0, nums)
    return list(new_seq)

print(keep_evens([3, 4, 6, 7, 0, 1]))
```

```
[4, 6, 0]
```

Activity: 2 -- ActiveCode (ac21_3_2)

Check Your Understanding

1. Write code to assign to the variable `filter_testing` all the elements in lst_check that have a w in them using filter.

Save & Run 5/14/2021, 11:12:27 PM - 2 of 2 Show in CodeLens

```

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

filter_testing = filter(lambda word: 'w' in word, lst_check)
```

Activity: 3 -- ActiveCode (ac21_3_3)

Result	Actual Value	Expected Value	Notes

Pass	['wat...ies']	['wat...ies']	Testing that filter_testing has the correct values.	Expand Differences
Pass	'map('	'\nlst_...heck)'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'filter('	'\nlst_...heck)'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'sum('	'\nlst_...heck)'	Testing your code (Don't worry about actual and expected values).	Expand Differences
Pass	'zip('	'\nlst_...heck)'	Testing your code (Don't worry about actual and expected values).	Expand Differences

You passed: 100.0% of the tests

2. Using filter, filter `lst` so that it only contains words containing the letter "o". Assign to variable `lst2`. Do not hardcode this.

Save & Run 5/14/2021, 11:04:59 PM - 2 of 2 Show in CodeLens

```

lst = ["witch", "halloween", "pumpkin", "cat", "candy", "wagon", "moon"]

lst2 = filter(lambda word: "o" in word, lst)
```

Activity: 4 -- ActiveCode (ac21_3_4)

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

You passed: 100.0% of the tests

You have attempted 5 of 4 activities on this page

23.2. Map">

23.4. List Comprehensions">

23.2. Map

✔ Completed. Well Done!

23.4. List Comprehensions">Next Section - 23.4. List Comprehensions

| Back to top