Iteration Patterns

CPE101 Winter 2019
@ Cal Poly SLO
By
Toshi

Learning Objectives

1. Iteration
    a. while -- nothing special
    b. For
        i. A good opportunity to introduce iteration over elements of a list, iteration over the indices of the list, and the use of enumerate to get both.
2. Patterns
    a. map, filter, fold -- Introduce patterns and functions (or comprehensions) in Python. The provided functions require passing functions as arguments which is technically optional.

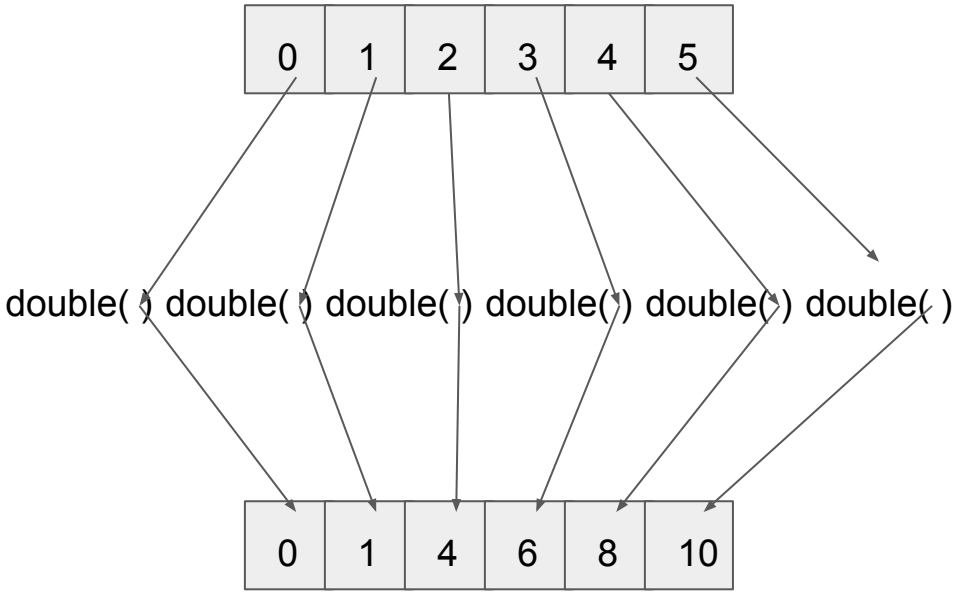
Map Pattern

Map a function to each element of a list

```
def double(n):
    return n * 2

def my_map(func, lst):
    result = []
    for n in lst:
        result.append(func(n))
    return result

numbers = [0, 1, 2, 3, 4, 5]
doubled = my_map(double, numbers)
print doubled # [0, 1, 4, 6, 8, 10]
```



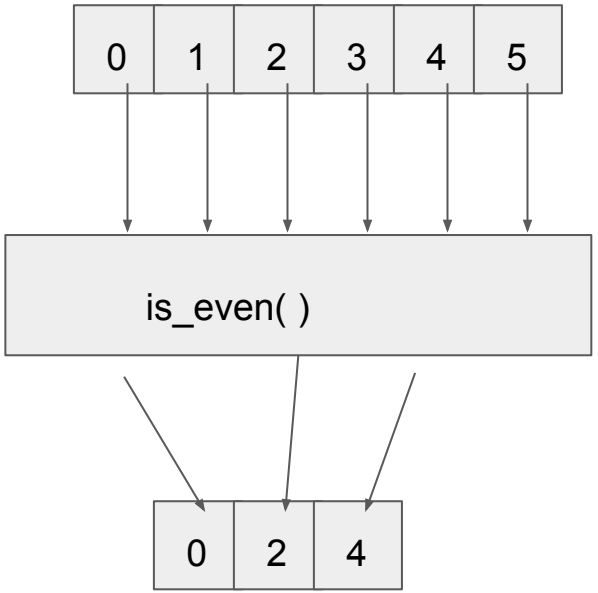
Filter Pattern

Filter elements of a list by a function

```
def is_even(n):
    return n % 2 == 0

def my_filter(func, lst):
    result = []
    for n in lst:
        if func(n):
            result.append(n)
    return result

numbers = [0, 1, 2, 3, 4, 5]
evens = my_filter(is_even, numbers)
print evens # [0, 2, 4]
```



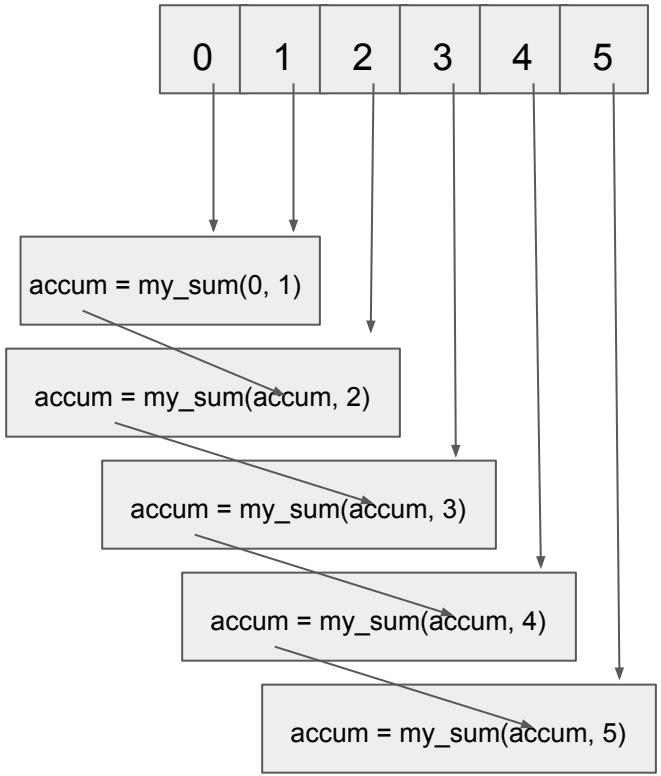
Reduce (Fold) Pattern

Reduce a list of values to one value

```
def my_sum(accum, n):
    return accum + n

def my_reduce(func, lst):
    accum = lst[0]
    for i in range(1, len(lst) ):
        accum = func(accum, lst[i])
    return accum

numbers = [0, 1, 2, 3, 4, 5]
total = my_reduce(my_sum, numbers)
print total # 15
```



Reduce

Python has a built-in function for summing numbers.

```
numbers = [0, 1, 2, 3, 4, 5]
total = sum(numbers)
print total # 15
```