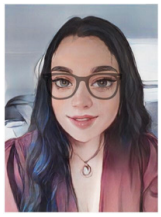
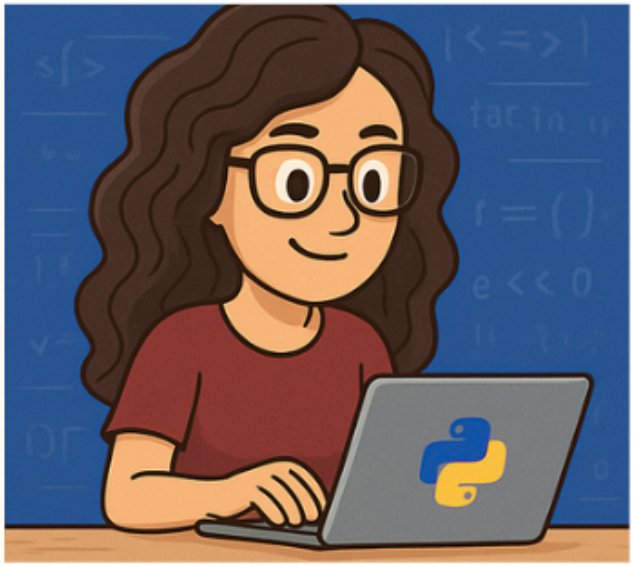


@pythonessdatadiaries

Python Filter and Lambda Exercises:

From Beginner to Advanced



This document provides Python exercises ,to enhance your functional programming skills focusing on,

- ***filter ( )***
- ***lambda functions***

Table of Contents

1. Beginner Exercises:
   - Exercise 1: Filter Even and Odd Numbers
   - Exercise 2: Filter Strings by Length
2. Intermediate Exercises:
   - Exercise 3: Filter Students by Height and Weight
   - Exercise 4: Filter Numbers with Sum of Digits > 0
3. Advanced Exercises:
   - Exercise 5: Filter and Sort a List of Dictionaries
   - Exercise 6: Filter Non-Repeated Characters in a String
4. Reference : w3resource

Exercise 1: Filter Even and Odd Numbers

Problem Statement:

Write a Python program to filter a list of integers using lambda to separate even and odd numbers.

0,2,4,6,8

1,3,5,7,9

Code:

```
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

# Filter even numbers
even_numbers = list(filter(lambda x: x % 2 == 0, numbers))
print("Even numbers:", even_numbers)

# Filter odd numbers
odd_numbers = list(filter(lambda x: x % 2 != 0, numbers))
print("Odd numbers:", odd_numbers)
```

Output:

```
Even numbers: [2, 4, 6, 8, 10]
Odd numbers: [1, 3, 5, 7, 9]
```

Explanation

- The filter function applies a lambda function to each element in the numbers list.
- For even numbers, lambda x: x % 2 == 0 returns True if the number is divisible by 2.
- For odd numbers, lambda x: x % 2 != 0 returns True if the number is not divisible by 2.
- The list() function converts the filter object to a list for display.

Exercise 2: Filter Strings by Length

Problem Statement

Write a Python program to filter a list of strings to find those with a length of 6 using lambda.

Code:

```
days = ["Monday", "Tuesday", "Friday", "Sunday", "Saturday"]
filtered_days = list(filter(lambda x: len(x) == 6, days))
print("Days with length 6:", filtered_days)
```

Output:

```
Days with length 6: ['Monday', 'Friday', 'Sunday']
```

Explanation

- The lambda x: len(x) == 6 function checks if each string in the days list has exactly 6 characters.
- The filter function returns only the strings that satisfy this condition.
- The result is converted to a list and printed display.

Exercise 3: Filter Students by Height and Weight

Problem Statement:

Write a Python program to filter a dictionary of students by height (> 6ft) and weight (> 70kg) using lambda.

Code:

```
students = {
    'Sanjana': (6.2, 71),
    'Aniruddha': (5.9, 65),
    'Siddharth': (6.0, 75),
    'Shobhna': (5.8, 66)
}

student_filtered = dict(filter(lambda item:item[1][0] > 6.0
                               and item[1][1] > 70,students.items()))
print("Height > 6ft and Weight> 70kg:",student_filtered)
```

Output:

```
Height > 6ft and Weight> 70kg: {'Sanjana': (6.2, 71)}
```

Explanation

- The students dictionary maps names to tuples of (height, weight).
- The lambda item: item[1][0] > 6.0 and item[1][1] > 70 function checks if the height (item[1][0]) is greater than 6.0 and weight (item[1][1]) is greater than 70.
- filter is applied to students.items(), and the result is converted back to a dictionary.

Exercise 4: Filter Numbers with Sum of Digits > 0

Problem Statement

Write a Python program to filter numbers in a list where the sum of digits is greater than 0, considering the first digit can be negative.

Code:

```
def sum_of_digits(num):
    sign = -1 if num <0 else 1
    num = str(abs(num))
    digit_sum = sum((int(digit) for digit in num))
    return digit_sum * sign

numbers = [110, -999, -3200, 78, 99]

filtered_numbers = list(filter(lambda x:sum_of_digits(x) > 0,numbers))
print("Numbers with sum of digits > 0:",filtered_numbers)
```

Output:

```
Numbers with sum of digits > 0: [110, 78, 99]
```

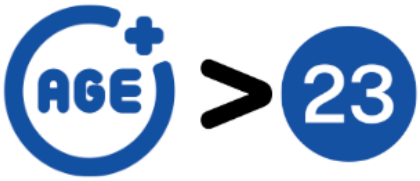
Explanation

- **Define the input list:** The list numbers = [110, -999, -3200, 78, 99] contains integers, some positive and some negative, to be filtered based on their digit sums.
- **sum_of_digits function**: Converts a number to its absolute value as a string, sums its digits, and multiplies by -1 if the original number was negative to preserve the sign.
- **Filter with lambda:** The filter(lambda x: sum_of_digits(x) > 0, numbers) keeps numbers where the sum of digits is positive, using sum_of_digits to compute the sum.
- **Convert and print:** Converts the filter object to a list with list() and prints numbers with a positive digit sum: [110, 78, 99].

Exercise 5: Filter and Sort a List of Dictionaries

Problem Statement

Write a Python program to filter a list of dictionaries by age (> 23) and sort by age using lambda.



Code:

```
people = [
    {'name': 'Summer', 'age': 25},
    {'name': 'Tanya', 'age': 22},
    {'name': 'Ankit', 'age': 28}
]

filter_by_sorted_age = sorted(list(filter(lambda x : x['age']>23, people)),
                              key = lambda x: x['age'])

print("People with age > 23, sorted by age:\n", filter_by_sorted_age)
```

Output:

```
People with age > 23, sorted by age:
 [{'name': 'Summer', 'age': 25}, {'name': 'Ankit', 'age': 28}]
```

Explanation

- The filter(lambda x: x['age'] > 23, people) selects dictionaries where the age is greater than 23.
- The sorted function uses lambda x: x['age'] to sort the filtered list by the 'age' key.
- The result is a sorted list of dictionaries.

Exercise 6: Filter Non-Repeated Characters in a String

Problem Statement

Write a Python program to find the first non-repeated character in a string using lambda and filter.

Code:

```
s = "Bubble"

non_repeated_char = next(filter(lambda x:s.count(x.lower()) == 1, s),None)

print("First non-repeated character:", non_repeated_char)
```

Output:

```
First non-repeated character: u
```

Explanation

- The lambda x: s.count(x) == 1 function checks if a character appears exactly once in the string.
- The filter function applies this condition to each character in the string.
- The next function returns the first character that satisfies the condition, or None if none exists.

References

- w3resource Python Lambda Exercises:
- w3resource Mastering Python 100 Exercises
- w3resource Python Programming Puzzles