ASSIGNMENT-3

https://github.com/arnoldrnld/ustglob/tree/main/assignments

Map (5 Coding Questions)

1. **Write a Python program using map() to convert a list of integers into their squares.**

```
nums = [1,2,3,4]
squares = list(map(lambda x:x**2, nums))
print(squares)
```

2. **Write a program using map() to convert all strings in a list to uppercase.**

```
l1 = ['Ball', 'Cat', 'Bat']
l2 = list(map(lambda x:x.upper(), l1))
print(l2)
```

3. **Given a list of temperatures in Celsius, use map() to convert them to Fahrenheit.**

```
celsius = [30, 32, 34]
farenheit = list(map(lambda c:c * (9/5) + 32, celsius))
print(farenheit)
```

4. **Write a program using map() to calculate the length of each word in a list of strings.**

```
l1 = ['Bat', 'Umbrella', 'Seahorse', 'Beach']
l2 = list(map(lambda x:len(x), l1))
print(l2)
```

5. **Use map() to add 10 to each element of a given list of numbers.**

```
nums = [30, 45, 78, 91, -32]
new_nums = list(map(lambda x:x+10, nums))
print(new_nums)
```

Filter (5 Coding Questions)

1. **Write a Python program using filter() to extract all even numbers from a list.**

```
nums = [30, 45, 78, 91, -32, ]
even_nums = list(filter(lambda x:x%2==0, nums))
print(even_nums)
```

2. **Write a program using filter() to select all words from a list that start with a vowel.**

```
l1 = ['Bat', 'Eat', 'Run', 'Apple', 'Orange']
l2 = list(filter(lambda x:x.lower()[0] in ('a','e','i','o','u'), l1))
print(l2)
```

3. **Given a list of integers, use filter() to remove all negative numbers.**

```
nums = [0, -1, 45, 78, -91, -32, ]
positives = list(filter(lambda x:x>0, nums))
print(positives)
```

4. **Write a program using filter() to find numbers greater than 50 from a list.**

```
nums = [0, -1, 45, 78, 91, -32, ]
greater = list(filter(lambda x:x>50, nums))
print(greater)
```

5. **Use filter() to extract all palindromic strings from a list.**

```
l1 = ['Cccbbccc', 'Malayalam', 'Run', 'Ababa', 'Orange']

def palindrome(word):
return word.lower() == word.lower()[::-1]

l2 = list(filter(lambda x:palindrome(x), l1))
print(l2)
```