filter() Function

In [ ]:

```

```

Decorator

In [ ]:

```python
def testing(fun):
    print("Hey! I am testing function")
    fun()

def test2():
    print("function test2 is executed")

testing(test2)
```

In [ ]:

```python
def testing1():
    print('Testing - 1')
    def testing2():
        print("Testing - 2")
        # def testing3():
        #     print("Testing - 3")
        # return testing3()
    return testing2

output = testing1()
output()
```

In [ ]:

```python
def wraper(fun): # Decorator defining
    def inner_function():
        print('Modification before calling function')
        fun()
        print("Modification after calling function")
    return inner_function

@wraper  # Decorator calling and changing functionality of the to_modify()
def to_modify():
    print("Function to be modified")

to_modify()

# output = wraper(to_modify)
# output()
```

In [ ]:

```python
def wrapper(fn):
    def inner_function(ls):
        total_sum = sum(ls)
        count = len(ls)
        fn(total_sum,count)
    return inner_function

@wrapper
def average_finder(total_sum,count):
    print(f"Your average is: {total_sum/count:.2f}")
```

```
ls= [10,20,30,40]
average_finder(ls)
```

Iterator

In [5]:
```
ls = [10,20,30,40,50]
# print(type(ls))
# print(type(iter(ls)))
var2 = iter(ls)
var2
```

```
3
```

In [ ]:
```
# next(ls)
```

In [ ]:
```
print(type(var2))

for item in var2:
    print(item)

print(type(var2))
```

In [ ]:
```
next(var2)
```

Generator

```
    It is a special type of function which uses yield keyword
    instead of return keyword and generator are memory efficient and
    returns generator object
```

In [8]:
```
def test(n):
    ls = []
    for i in range(n):
        ls.append(i)
    return ls

test(12)
```

Out[8]:
```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11]
```

In [14]:
```
def test2(n):
    for i in range(n):
        yield i # it stores the local variables
```

In [33]:
```
output = test2(500)
output
```

Out[33]:
```
<generator object test2 at 0x000001DD7F6F4930>
```

In [34]:
```
total = 0
for item in output:
    total += item
total
```

```
Out[34]: 124750
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```