**Decorator Task 1:**

**Definition: Decorator with argument**

**Code:**

# Decorator for Upper Case

def make\_upper(fn):

    def wrapper(String):

        return string.upper()

    return wrapper

@make\_upper

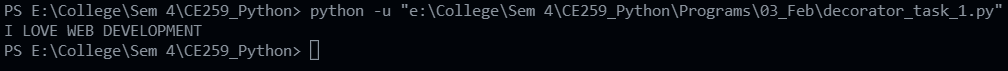
def hello(string):

    return string

string = 'I Love Web Development'

print(hello(string))

**Output:**



**Map Task (Lambda):**

**Definition: Use Map with Lambda Function**

**Code:**

item\_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

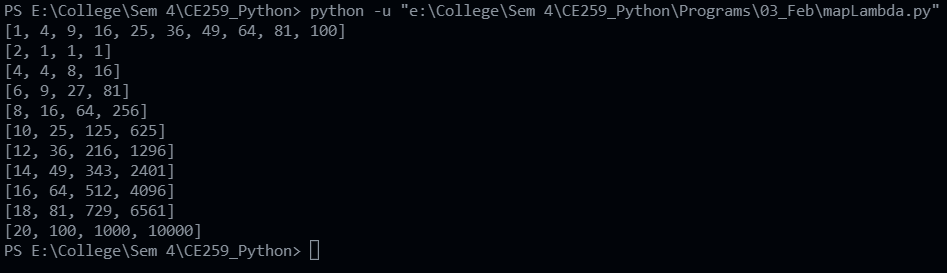
square = list(map(lambda x: x\*\*2, item\_list))

print(square)

for i in item\_list:

    print(list(map(lambda function:function(i), [lambda x: x+x, lambda x: x\*\*2, lambda x: x\*\*3, lambda x: x\*\*4])))

**Output:**



**Map Task 1:**

**Definition: Given a list of fruits, use map function and make two lists (FruitsStartingWithA, FruitsStartingWithE), and append the corresponding items into both lists.**

**Code:**

fruits = ['apple', 'banana', 'cherry', 'durian', 'fig', 'grape', 'honeydew', 'jackfruit', 'kiwi', 'lemon', 'mango', 'nectarine', 'orange', 'papaya', 'peach', 'pear', 'pineapple', 'plum', 'pomegranate', 'quince', 'raspberry', 'strawberry', 'tangerine', 'watermelon']

fruits\_starting\_with\_a = []

fruits\_ending\_with\_e = []

def distribute\_fruits(fruit):

        if fruit[0] == 'a':

            fruits\_starting\_with\_a.append(fruit)

        if fruit[-1] == 'e':

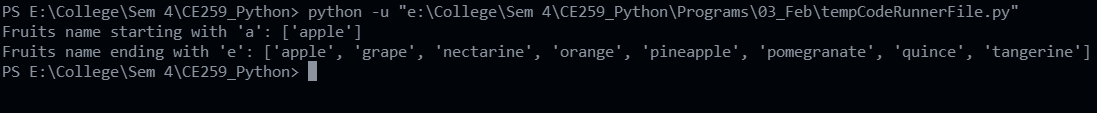
            fruits\_ending\_with\_e.append(fruit)

list(map(distribute\_fruits, fruits))

print(f"Fruits name starting with 'a': {fruits\_starting\_with\_a}")

print(f"Fruits name ending with 'e': {fruits\_ending\_with\_e}")

**Output:**



**Map Task 2:**

**Definition: Make round\_off function using map function, also give attributes(upper and lower bound of the list of rounding off).**

**Code:**

circle\_area = [23.145, 25.894, 63.857, 42.857, 12.567, 18.857, 23.145, 25.894, 63.857, 42.857, 12.567, 18.857]

a = int(input("Enter the lower bound of the list: "))

b = int(input("Enter the upper bound of the list: "))

c = int(input("Enter the number of decimal places: "))

def round\_off(x):

    return round(x, c)

print(list(map(round\_off, circle\_area[a:b])))

**Output:**

