**Assignment**

**1.** Write a Python program to create a lambda function that adds 15 to a given number passed in as an argument, also create a lambda function that multiplies argument x with argument y and prints the result.  
Sample Output:  
25  
48

add\_15 = lambda x: x + 15

print(add\_15(10))

multiply = lambda x, y: x \* y

print(multiply(6, 8))

**2.** Write a Python program to create a function that takes one argument, and that argument will be multiplied with an unknown given number.  
Sample Output:  
Double the number of 15 = 30  
Triple the number of 15 = 45  
Quadruple the number of 15 = 60  
Quintuple the number 15 = 75  
def multiplier(n):

return lambda x: x \* n

double = multiplier(2)

triple = multiplier(3)

quadruple = multiplier(4)

quintuple = multiplier(5)

print(f"Double the number of 15 = {double(15)}")

print(f"Triple the number of 15 = {triple(15)}")

print(f"Quadruple the number of 15 = {quadruple(15)}")

print(f"Quintuple the number of 15 = {quintuple(15)}")

**3.** Write a Python program to sort a list of tuples using Lambda.  
Original list of tuples:  
[('English', 88), ('Science', 90), ('Maths', 97), ('Social sciences', 82)]  
Sorting the List of Tuples:  
[('Social sciences', 82), ('English', 88), ('Science', 90), ('Maths', 97)]

tuples = [('English', 88), ('Science', 90), ('Maths', 97), ('Social sciences', 82)]

sorted\_tuples = sorted(tuples, key=lambda x: x[1])

print(sorted\_tuples)

**4.** Write a Python program to sort a list of dictionaries using Lambda.  
Original list of dictionaries :  
[{'make': 'Nokia', 'model': 216, 'color': 'Black'}, {'make': 'Mi Max', 'model': '2', 'color': 'Gold'}, {'make': 'Samsung', 'model': 7, 'color': 'Blue'}]  
Sorting the List of dictionaries :  
[{'make': 'Nokia', 'model': 216, 'color': 'Black'}, {'make': 'Samsung', 'model': 7, 'color': 'Blue'}, {'make': 'Mi Max', 'model': '2', 'color': 'Gold'}]

dictionaries = [

{'make': 'Nokia', 'model': 216, 'color': 'Black'},

{'make': 'Mi Max', 'model': '2', 'color': 'Gold'},

{'make': 'Samsung', 'model': 7, 'color': 'Blue'}

]

sorted\_dictionaries = sorted(dictionaries, key=lambda x: int(x['model']))

print(sorted\_dictionaries)

**5.** Write a Python program to filter a list of integers using Lambda.  
Original list of integers:  
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
Even numbers from the said list:  
[2, 4, 6, 8, 10]  
Odd numbers from the said list:  
[1, 3, 5, 7, 9]

# List of integers

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 from the said list:", even\_numbers)

# Filter odd numbers

odd\_numbers = list(filter(lambda x: x % 2 != 0, numbers))

print("Odd numbers from the said list:", odd\_numbers)

**6.** Write a Python program to square and cube every number in a given list of integers using Lambda.  
Original list of integers:  
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
Square every number of the said list:  
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]  
Cube every number of the said list:  
[1, 8, 27, 64, 125, 216, 343, 512, 729, 1000]

# List of integers

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

# Square every number

squares = list(map(lambda x: x \*\* 2, numbers))

print("Square every number of the said list:", squares)

# Cube every number

cubes = list(map(lambda x: x \*\* 3, numbers))

print("Cube every number of the said list:", cubes)

**7.** Write a Python program to find if a given string starts with a given character using Lambda.  
Sample Output:  
True  
False

# Check if a string starts with a specific character

starts\_with = lambda s, c: s.startswith(c)

print(starts\_with("apple", "a")) # True

print(starts\_with("banana", "b")) # True

print(starts\_with("apple", "b")) # False

**8.** Write a Python program to extract year, month, date and time using Lambda.  
Sample Output:  
2020-01-15 09:03:32.744178  
2020  
1  
15  
09:03:32.744178

from datetime import datetime

# Current date and time

now = datetime.now()

print(now)

# Extract year, month, day, time

year = lambda x: x.year

month = lambda x: x.month

day = lambda x: x.day

time = lambda x: x.time()

print(year(now))

print(month(now))

print(day(now))

print(time(now))

**9.** Write a Python program to check whether a given string is a number or not using Lambda.  
Sample Output:  
True  
True  
False  
True  
False  
True  
Print checking numbers:  
True  
True

# Check if a string is numeric

is\_number = lambda x: x.replace('.', '', 1).isdigit()

print(is\_number("123")) # True

print(is\_number("12.3")) # True

print(is\_number("abc")) # False

print(is\_number("0.5")) # True

print(is\_number("-123")) # False

print(is\_number("7e2")) # False

**10.** Write a Python program to create Fibonacci series up to n using Lambda.  
Fibonacci series upto 2:  
[0, 1]  
Fibonacci series upto 5:  
[0, 1, 1, 2, 3]  
Fibonacci series upto 6:  
[0, 1, 1, 2, 3, 5]  
Fibonacci series upto 9:  
[0, 1, 1, 2, 3, 5, 8, 13, 21]

from functools import reduce

# Fibonacci series generator

fibonacci = lambda n: reduce(lambda x, \_: x + [x[-1] + x[-2]], range(n-2), [0, 1])

print("Fibonacci series up to 2:", fibonacci(2))

print("Fibonacci series up to 5:", fibonacci(5))

print("Fibonacci series up to 6:", fibonacci(6))

print("Fibonacci series up to 9:", fibonacci(9))

**11.** Write a Python program to find the intersection of two given arrays using Lambda.  
Original arrays:  
[1, 2, 3, 5, 7, 8, 9, 10]  
[1, 2, 4, 8, 9]  
Intersection of the said arrays: [1, 2, 8, 9]

# Two arrays

arr1 = [1, 2, 3, 5, 7, 8, 9, 10]

arr2 = [1, 2, 4, 8, 9]

# Find the intersection

intersection = list(filter(lambda x: x in arr2, arr1))

print("Intersection of the arrays:", intersection)