**TASK FOUR**

**TRADITIONAL FUNCTIONS,ANONYMOUS FUNCTIONS &**

**HIGHER ORDER FUNCTIONS**

**1.** Write a program to reverse a string.

**Sample input:** “1234abcd”

**Expected output:** “dcba4321”

input\_value = "1234abcd"

def reverse\_string(input\_value):

    return input\_value[::-1]

print(reverse\_string(input\_value))

OUTPUT:

dcba4321

**2.** Write a function that accepts a string and prints the number of uppercase letters and lowercase letters.

**Sample input:** “abcSdefPghijQkl”

**Expected Output:** No. of Uppercase characters : 3 No. of Lower case Characters : 12

input\_value = "abcSdefPghijQkl"

upper\_count = 0

lower\_count = 0

def letter():

    global upper\_count

    global lower\_count

    for i in input\_value:

        if i.isupper():

            upper\_count += 1

        else:

            lower\_count += 1

letter()

print("No. of Uppercase characters : ", upper\_count)

print("No. of Lowercase Characters : ", lower\_count)

OUTPUT:

No. of Uppercase characters : 3

No. of Lowercase Characters : 12

**3.** Create a function that takes a list and returns a new list with unique elements of the first list.

list\_1 = [2,3,2,4,5,4,5,]

def unique\_list(list\_1):

    u\_list= set(list\_1)

    return list(u\_list)

print(unique\_list(list\_1))

OUTPUT:

[2, 3, 4, 5]

**4.** Write a program that accepts a hyphen-separated sequence of words as input and prints the words in a hyphen-separated sequence after sorting them alphabetically.

data\_hypen = input("Enter a hypen-seperated sequence of words :")

hypen\_words =data\_hypen.split('-')

hypen\_words.sort()

sorted\_hypen\_word = ('-'.join(hypen\_words))

print(sorted\_hypen\_word)

OUTPUT:

Enter a hypen-seperated sequence of words :Q-W-E-R-T-Y-W-O-R-L-D

D-E-L-O-Q-R-R-T-W-W-Y

**5.** Write a program that accepts a sequence of lines as input and prints the lines after making all characters in the sentence capitalized.

**Sample input:** Hello world Practice makes man perfect

**Expected output:** HELLO WORLD PRACTICE MAKES MAN PERFECT

sequence\_lines = input("Enter  sequence of lines:")

for i in sequence\_lines:

    cap\_char= i.capitalize()

    print(cap\_char, end = "")

OUTPUT:

Enter sequence of lines:Hello world Practice makes man perfect

HELLO WORLD PRACTICE MAKES MAN PERFECT

**6.** Define a function that can receive two integral numbers in string form and compute their sum and print it in the console.

num1 = input("Enter first value:")

num2 = input("Enter second value:")

def cal\_sum(a, b):

    s = int(a) + int(b)

    return s

total= cal\_sum(num1, num2)

print("Sum of two values :", total)

OUTPUT:

Enter first value:4

Enter second value:8

Sum of two values : 12

**7.** Define a function that can accept two strings as input and print the string with the maximum length in the console. If two strings have the same length, then the function should print both the strings line by line.

a = input('Enter the first name:')

b = input('Enter the second name:')

def  max\_len(a,b):

    if len(a)== len(b):

        print(a,b ,sep='\n')

    elif len(a) > len(b):

        print(a)

    else:

        print(b)

max\_len(a,b)

OUTPUT:

Enter the first name:RONAK

Enter the second name:CHANDGADHIA

CHANDGADHIA

**8.** Define a function which can generate and print a tuple where the values are square of numbers between 1 and 20 (both 1 and 20 included).

def square\_num(n):

    list\_1 = []

    for i in range(1,n+1):

        a= i\*i

        list\_1.append(a)

    return(list\_1)

print(tuple(square\_num(30)))

OUTPUT:

(1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400, 441, 484, 529, 576, 625, 676, 729, 784, 841, 900)

**9.** Write a function called showNumbers that takes a parameter called limit. It should print all the numbers between 0 and limit with a label to identify the even and odd numbers.

**Sample input:** show Numbers(3) (where limit=3)

**Expected output:**

0 EVEN

1 ODD

2 EVEN

3 ODD

def showNumbers(limit):

    for i in range(limit):

        if i % 2 == 0:

            print(i, end=" ")

            print("EVEN")

        else:

            print(i, end=" ")

            print("ODD")

showNumbers(4)

OUTPUT:

0 EVEN

1 ODD

2 EVEN

3 ODD

**10.** Write a program which uses filter() to make a list whose elements are even numbers between 1 and 20 (both included)

list\_1 = range(1, 21)

print(list(filter(lambda x: x % 2 == 0, list\_1)))

OUTPUT:

[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]

**11.** Write a program which uses map() and filter() to make a list whose elements are squares of even numbers in [1,2,3,4,5,6,7,8,9,10].

Hints: Use filter() to filter even elements of the given listUse map() to generate a list of squares of the numbers in the filtered list. Use lambda() to define anonymous functions.

list\_1 = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

print(list(map(lambda x: x \* x,(filter(lambda x: x % 2 == 0, list\_1) ))))

OUTPUT:

[4, 16, 36, 64, 100]

**12.** Write a function to compute 5/0 and use try/except to catch the exceptions.

def div\_n(n):

    try:

       k = n/0

    except ZeroDivisionError:

        print("Can't divide by zero")

    finally:

        print("This is finally executed")

div\_n(5)

OUTPUT:

Can't divide by zero

This is finally executed

**13.** Flatten the list [1,2,3,4,5,6,7] into 1234567 using reduce().

from functools import reduce

flatten\_list = [1,2,3,4,5,6,7]

a=reduce(lambda total, d: 10 \* total + d, flatten\_list)

print(a)

OUTPUT:

1234567

**14.** Write a program in Python to find the values which are not divisible by 3 but are a multiple of 7. Make sure to use only higher order functions.

def div(n):

    for i in range(n):

        if (n % 3 != 0) & (n % 7 == 0):

            return (n)

list\_1 = range(1, 21)

a = (list(map(div, list\_1)))

print(a)

OUTPUT:

[None, None, None, None, None, None, 7, None, None, None, None, None, None, 14, None, None, None, None, None, None]

**15**. Write a program in Python to multiply the elements of a list by itself using a traditional function and pass the function to map() to complete the operation.

def mulSqaure(n):

     return(n\*n)

list\_1 = range(1,10)

print(list(map(mulSqaure,list\_1)))

OUTPUT:

[1, 4, 9, 16, 25, 36, 49, 64, 81]

**16.** What is the output of the following codes:

**(i)** def foo():

try:

return 1

finally:

return 2

k = foo()

print(k)

OUTPUT:

2

**(ii)** def a():

try:

f(x, 4)

finally:

print('after f')

print('after f?')

a()

OUTPUT:

2

after f

Traceback (most recent call last):

File "D:\Python\Task 4\Task4\_Q16.py", line 21, in <module>

a()

File "D:\Python\Task 4\Task4\_Q16.py", line 16, in a

f(x, 4)

NameError: name 'f' is not defined

**END OF TASK FOUR**