**Python Lab Programs on functions:**

1. **Write a python program to test if a word contains three consecutive double letters.**

def is\_triple\_double(word):

i = 0

count = 0

while i < len(word)-1:

if word[i] == word[i+1]:

count = count + 1

if count == 3:

return True

i = i + 2

else:

i = i + 1

return False

def find\_triple\_double():

fin = input("Enter a word: ")

res=is\_triple\_double(fin)

if res:

print("Three consecutive double letter found in word:",fin)

else:

print("Three consecutive double letter not found in word:",fin)

find\_triple\_double()

1. **Write a Python program to get the n (non-negative integer) copies of the first 2 characters of a given string. Return the n copies of the whole string if the length is less than 2.**

def substring\_copy(str, n):

flen = 2

if flen > len(str):

flen = len(str)

substr = str[:flen]

result = ""

for i in range(n):

result = result + substr

return result

print(substring\_copy('abcdef', 2))

print(substring\_copy('p', 3));

1. **Write a Python program for binary search**

def binary\_search(item\_list,key):

low=0

high=len(item\_list)-1

while(low<=high):

mid=(low+high)//2

if(item\_list[mid]==key):

return mid

else:

if(item\_list[mid]<key):

low=mid+1

else:

high=mid-1

return -1

result=binary\_search([2,4,5,6,7],6)

if(result!=-1):

print("Key present at position",result+1)

else:

print("key not found")

1. **Write a python program to create a simple graphical triangle importing Turtle and fill the triangles with different colours.**

import turtle

t = turtle.Turtle()

pen\_color = input("enetr pencolor:")

fill\_color = input("enter fillcolor:")

t.color(pen\_color,fill\_color)

t.begin\_fill()

for i in range(3):

t.fd(100)

t.lt(120)

t.end\_fill()

t.pu()

t.lt(135)

t.fd(100)

t.pd()

t.begin\_fill()

for i in range(3):

t.fd(100)

t.lt(120)

t.end\_fill()

1. Write a program to print the Fibonacci series using recursion.

def fibonacci(n):

if(n<2):

return 1

else:

return (fibonacci(n-1)+fibonacci(n-2))

n=int(input("Enter the number of terms:"))

print("Fibonacci series:")

for i in range(n):

print(fibonacci(i))

**Programs on Lists**

1. **Write a python program to demonstrate the following.**

1. **Find list1[2:5],list1[::2],list1[1::3] for the given list list1= [1,‘a’,”Hello”, [2,3,4,5,6], 4.5]**
2. **Append new values in the list**
3. **Remove existing values in the list**
4. **Insert a list in another list**
5. **Check if a element is present or not in a list**
6. **Concatenate two lists**
7. **Insert a element at a specified index in the list**
8. **Reverse the elements in the list**
9. **Print list of cubes using list comprehensive**

#Find list1[2:5],list1[::2],list1[1::3] for the given list

# list1= [1,‘a’,”Hello”, [2,3,4,5,6], 4.5]

list1= [1,"a","Hello", [2,3,4,5,6], 4.5]

print(list1[2:5],list1[::2],list1[1::3])

#Append new values in the list

list1.append(200)

print(list1)

# Remove existing values in the list

list1.remove("Hello")

print(list1)

#Insert a list in another list

list1.insert(3,[2.2,1,"Hai"])

print(list1)

#Check if a element is present or not in a list

res="a" in list1

print(res)

res1="Good" not in list1

print(res1)

res1=200 not in list1

print(res1)

print(list1)

#Concatenate two lists

list2=[10,20,30,40]

list1=list1+list2

print(list1)

#Insert a element at a specified index in the list

list1.insert(3,"Good Morning")

print(list1)

#Reverse the elements in the list

list1.reverse()

print(list1)

#Print list of cubes using list comprehensive

cubes=[i\*\*3 for i in range(10)]

print(cubes)

1. **Write a python program to remove all duplicates from a list.**

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

print("Original List :", num\_list)

i=0

while i<len(num\_list):

num=num\_list[i]

for j in range(i+1, len(num\_list)):

val= num\_list[j]

if val== num:

num\_list.pop(j)

i=i+1

print("list after removing duplicates : ", num\_list)

1. **Write a python program that convert list of temperatures in Celsius into Fahrenheit.**

def convert\_to\_F(Temp\_C):

return ((float(9)/5)\*Temp\_C + 32)

Temp\_in\_C = (36.5, 37, 37.5,39)

Temp\_in\_F = list(map(convert\_to\_F, Temp\_in\_C))

print("List of temperatures in Celsius : ", Temp\_in\_C)

print("List of temperatures in Fahrenheit : ", Temp\_in\_F)

1. **Write a program to the following using reduce() function**
   1. **Find the largest number in a list**
   2. **Find addition of the values in a list**

import functools

def max\_ele(x,y):

if(x>y):

return x

else:

return y

num\_list=[2,3,1,10,6,2]

print("Largest value in the list is:",functools.reduce(max\_ele,num\_list))

def add(x,y):

return x+y

num\_list1= [4, 3, 1, 10, 6, 2]

print ("sum value in the list is:",functools.reduce(add,num\_list1))

**Programs on Tuple**

1. Write a program to separate an even and odd numbers in a given tuple.

Tup=(2,3,4,5,6,7,8)

evenTup=()

oddTup=()

for i in Tup:

if i%2==0:

evenTup=evenTup+(i,)

else:

oddTup=oddTup+(i,)

print("Even Elements Tuple:",evenTup)

print("odd Elements Tuple:",oddTup)

**Programs on sets:**

1. Write a program that generates a set of prime numbers and another set of odd numbers. Demonstrate the result of union, intersection, difference, and symmetric difference operations on these sets.

odds = set([x\*2+1 for x in range(1,10)])

print(odds,end='\n')

primes=set()

for i in range(2, 20):

j=2

flag = 0

while j<(i//2):

if i%j ==0:

flag = 1

j+=1

if flag==0:

primes.add(i)

print(primes,end='\n')

print("UNION : ",odds.union(primes))

print("INTERSECTION : ", odds.intersection(primes))

print("SYMMETRIC DIFFERENCE : ", odds.symmetric\_difference(primes))

print("DIFFERENCE : ", odds.difference(primes))

program on Dictionary

1. Write a program that has dictionary of names of students and a list of their marks in 4 subjects. Create another dictionary from this dictionary that has name of the students and their total marks. Find out the topper and his/her score.

Marks ={ 'Neha' : [97,89,94, 90], 'Mitul' :[92,91,94,87], 'Shefali' : [67,99, 88,90]}

tot=0

Tot\_Marks = Marks.copy()

for key, val in Marks.items():

tot = sum(val)

Tot\_Marks[key] = tot

print(Tot\_Marks)

max = 0

Topper = ' '

for key, val in Tot\_Marks.items():

if val>max:

max = val

Topper = key

print("Topper is : ", Topper, "with marks =" , max)