**Practical No: 4**

1. To check whether string is palindrome or not using function recursion

Code:

def isPalindrome(string):

if len(string)<=1:

return True

else:

return string[0]==string[-1] and isPalindrome(string[1:-1])

while True:

print("\nPress 9 to quit.\n")

s = input("Enter a string to check if it's Palindrome or not\n");

s = s.lower();

if s=="9":

break

elif(isPalindrome(s) == True):

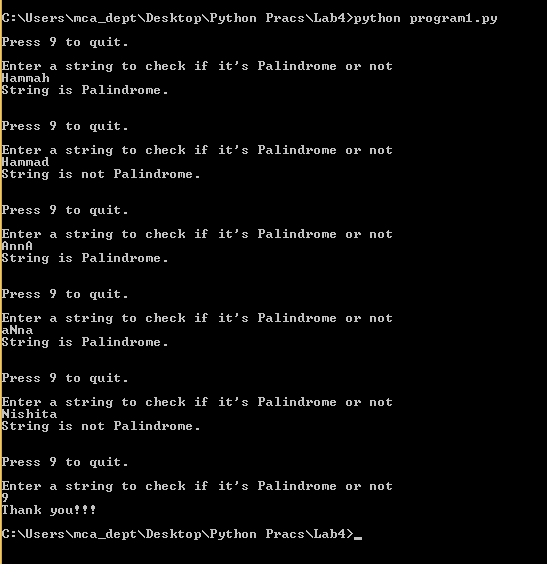
print("String is Palindrome.\n");

else:

print("String is not Palindrome.\n");

print("Thank you!!!")

Output:



1. To find Fibonacci series using recursion

Code:

fibonacci\_cache = {}

def fibonacci(n):

if n in fibonacci\_cache:

return fibonacci\_cache[n]

if n==1:

value = 1

elif n==2:

value = 1

elif n>2:

value = fibonacci(n-2)+fibonacci(n-1)

fibonacci\_cache[n] = value

return value

while True:

print("\nPress 0 to quit.\n")

n = int(input("Enter range for fibonacci series:\n"))

if n==0:

break

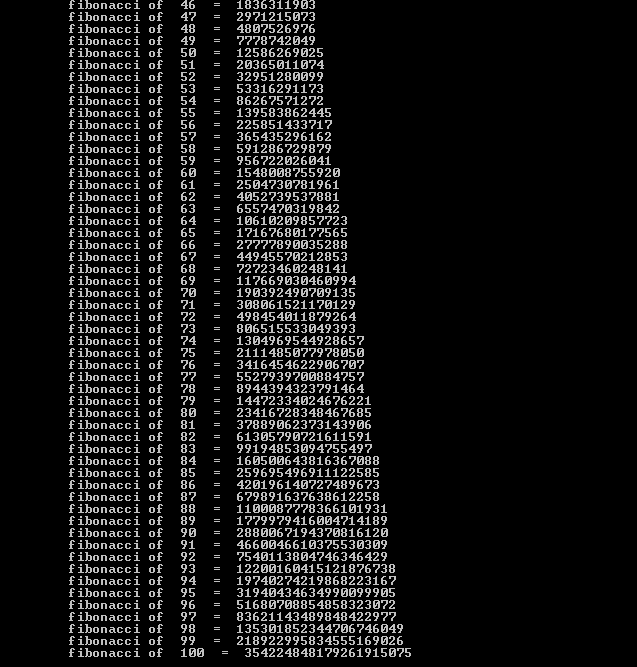
else:

for i in range(1,n+1):

print("\t fibonacci of ",i," = ",fibonacci(i))

print("Thank you!!!")

Output:



1. To find binary equivalent of number using recursion

Code:

Binary\_Storage = []

count = 0

def convertB(n,count):

if(n>0):

count=count+1

Binary\_Storage.append(n%2)

convertB(n//2,count)

while True:

print("\nPress 0 to quit.\n")

s = int(input("Enter a decimal number to convert it into binary number:\n"))

if s==0:

break

else:

convertB(s,-1)

Binary\_Storage.reverse()

print(\*Binary\_Storage)

Binary\_Storage.clear()

print("Thank you!!!")

Output:



1. To use lambda function on list to generate filtered list, mapped list and reduced list

Code:

from functools import reduce

class AnonDemo:

def lambdaDemo(self):

sum = lambda x, y : x + y

a=[2,3,5,6,7]

b=[6,9,2,1,8]

print(sum(a,b))

print("Sum of elements using lambda function :- ",sum(3,5))

def mapDemo(self):

items=[1,2,3,4,5]

squared = list(map(lambda x : x\*\*2, items))

print("Squared Items using map function :- ",squared)

def filterDemo(self):

number\_list = range(-10, 10)

less\_than\_zero = list(filter(lambda x: x < 0, number\_list))

print("Less than zero elements using filter function :- ",less\_than\_zero)

def reduceDemo(self):

product = reduce((lambda x, y: x \* y), [1, 2, 3, 4])

print("Product of all items using reduce function :- ",product)

ob = AnonDemo()

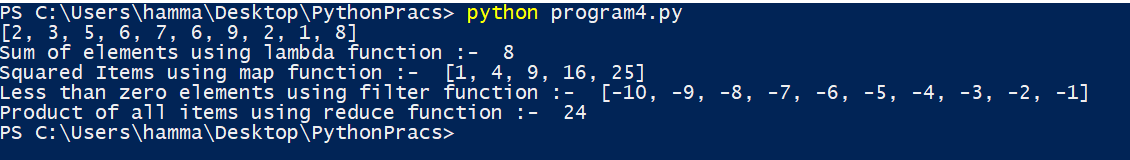
ob.lambdaDemo()

ob.mapDemo()

ob.filterDemo()

ob.reduceDemo()

Output:



1. Convert the temperature in Celsius to Fahrenheit in list using anonymous function

Code:

class Cf:

def convert(self):

n = int(input("Enter number of inputs :- \n"))

a = []

for i in range(0,n):

c = float(input("Enter celsius to convert :- \n"))

a.append(c)

return a

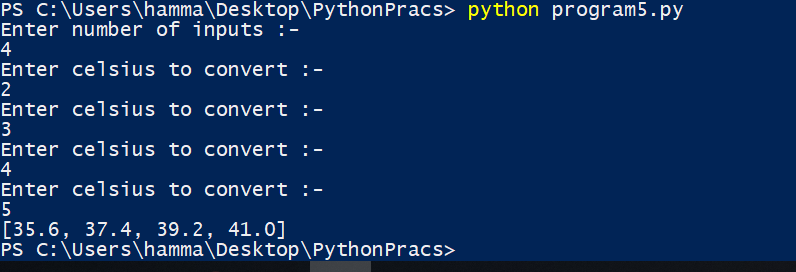
ob = Cf()

t = ob.convert()

f = list(map(lambda x: (float(9)/5)\*x + 32, t))

print(f)

Output:



1. To create module in python and access functions of the module by importing it to another file/module. (Calculator program)

Code:

#name of the main file is calc.py

import calculator

x = float(input("Enter first number: "))

y = float(input("Enter second number: "))

while True:

print("Select one of the operations \n1.Addition \n2.Subtract \n3.Multiplication \n4.Division \n5.Mod \n6.Exit")

select = int(input(" "))

if(select == 1):

print(calculator.Add(x, y))

elif(select == 2):

print(calculator.Sub(x, y))

elif(select == 3):

print(calculator.Mul(x, y))

elif(select == 4):

print(calculator.Div(x, y))

elif(select == 5):

print(calculator.Mod(x, y))

elif(select == 6):

break

else:

print("Invalid input")

file to be imported: calculator.py

def Add(a, b):

r = a + b

return r

def Sub(a, b):

r = a - b

return r

def Mul(a, b):

r = a \* b

return r

def Div(a, b):

if (a != 0 and b != 0):

r = a // b

return r

else:

return ("Divion is not possible , its not defined ")

def Mod(a, b):

r = a % b

return r

Output:

