Q1)**Aim: Write a program that asks the user for a weight in kilograms and converts it to pounds. There are 2.2 pounds in a kilogram.**

Program:

```
weight = float(input("Enter weight in kilograms: "))
pound=weight*2.2 print(weight,"Kilograms is equal
to ",pound,"Pounds")
```

Q2)**Aim: Write a program that asks the user to enter three numbers (use three separate input statements). Create variables called total and average that hold the sum and average of the three numbers and print out the values of total and average.**

Program:

```
n1 = int(input("Enter first number: ")) n2 =
int(input("Enter secod number: ")) n3 =
int(input("Enter third number: ")) total = n1
+ n2 + n3 average = total/3 print("Sum of
three numbers is:",total) print("Average of
three numbers is:",average)
```

Q3)**Aim: Write a program that uses a for loop to print the numbers 8, 11, 14, 17, 20, . . . , 83, 86, 89.**

Program:

```
n = int(input("Enter range: "))
for i in range(8,n,3):
print(i,end="") if i<n-3:
        print(", ",end="")
```

Q4)**Aim: Write a program that asks the user for their name and how many times to print it. The program should print out the user's name the specified number of times.**

Program:

```
name = input("Enter your name: ")
n = int(input("How many times you want to print? "))
for i in range(n): print(name)
```

Q5)**Aim: Use a forloop to print a triangle like the one below. Allow the user to specify how high the triangle should be Program:**

```
h = int(input("Enter height of triangle: "))
for i in range(h):
    for j in range(i+1):
        print("*",end="")
    print()
```

Q6)**Aim: Generate a random number between 1 and 10. Ask the user to guess the number and print a message based on whether they get it right or not.**

Program:

```
import random
sysgen = random.randint(1,10)
num = int(input("Guess a number between 1 and 10: "))
if num==sysgen:
    print("You guess is right.")
```

```
else: print("Sorry! System generated number
   is",sysgen)
```

Q7)**Aim: Write a program that asks the user for two numbers and prints Close if the numbers are within .001 of each other and Not close otherwise program:**

```
x=float(input('ENTER X'))
y=float(input('ENTER Y'))
z=x-y
temp=(round(z,3))
if temp<=0.001 :
   print('close')
else:
   print('not close')
```

(or)

Note: 'abs' is a built-in math function in python which is used for absolute value.

```
from decimal import *
num1 = Decimal(input("Enter number 1 : "))
num2 = Decimal(input("Enter number 2 : "))
diff=abs(num1-num2)
if (diff<=0.001):
print("close") else:
print("Not Close")
```

Q8)**Aim: Write a program that asks the user to enter a word and prints out whether that word contains any vowels Program:**

```
string=input('enter')
for char in string:
   if char in "aeiouAEIOU":
      print('yes') print(
      char,end=',')
```

Q9)**Aim:Write a program that asks the user to enter two strings of the same length. The program should then check to see if the strings are of the same length. If they are not, the program should print an appropriate message and exit. If they are of the same length, the program should alternate the characters of the two strings. For example, if the user enters abcdeandABCDE the program should print out AaBbCcDdEe.**

Program:

```
string1=input("enter string1")
string2=input("enter string2")
result=" " #temparary variable for storing data
if(len(string1)==len(string2)):
   for i in range(len(string1)):
      result+=string1[i]+string2[i]
   else:
      print("Alternate the characters of the two strings::",result)
else: print('Both string are Not
   Equal::')
```

Q10)**Aim: Write a program that asks the user for a large integer and inserts commas into it according to the standard American convention for commas in large numbers. For instance, if the user enters 1000000, the output should be 1,000,000.**

Program:

```
num=int(input('enter a long integer'))
st=str(num)#type conversion for i in
range(len(st)-3,0,-3):
   st=st[:i]+','+st[i:]
print("standard American convention format=",st)
```

Q11)**Aim: In algebraic expressions, the symbol for multiplication is often left out, as in 3x+4y or 3(x+5). Computers prefer those expressions to include the multiplication symbol, like 3*x+4*y or 3*(x+5). Write a program that asks the user for an algebraic expression and then inserts multiplication symbols where appropriate .**

Program:

```
exp=input('ENTER
EXPRESSION')
l=list(exp)
i=0
result=" while
i<len(l):
   if l[i]=='(':
      index=l.index(')')
   s2="".join(l[i:index+1])
   result=result+'*'+s2
   i+=len(s2) elif
   l[i].isalpha():
      result=result+'*'+l[i
   ] i+=1 else:
   result+=l[i]
      i+=1
print(result)
```

12) Write a program that generates a list of 20 random numbers between 1 and 100.

(a) Print the list.

(b) Print the average of the elements in the list.

(c) Print the largest and smallest values in the list.

(d) Print the second largest and second smallest entries in the list(e) Print how many even numbers are in the list.

```
import random

l=[] for num in

range(20):

   l.append(random.randint(1,100))
print("List Elements are:",l) avg=sum(l)/len(l)

print("The Average of the Elements in the list

is=",avg) print("Largest Value in the list is :",max(l))

print("Smallest Value in the list is :",min(l))
```

```
l1=sorted(l) print("second Largest value in the List
is:",l1[-2]) print("Second smallest  value in the List
is:",l1[1])
count=0
for i in l1:     if
(i%2==0):
       count=count+1 print("Number of Even Number
in the List is :",count)
```

Q13)**Write a program that asks the user for an integer and creates a list that consists of the factors of that integer. program:**

```
x = int(input("Enter Integer value"))
factorspos = []
print("The factors of",x,"are:",end='')
if x > 0: # if input is postive for i in
range(1,x+1):
       if x % i == 0:
          factorspos.append(i)
   print(factorspos)
elif x < 0: #if input is negative
   for i in range(x,0):
       if x % i == 0:
          factorspos.append(i)
   print(factorspos)
else: print('Not
   Possible')
```

Q14)**Write a program that generates 100 random integers that are either 0 or 1. Then find the longest run of zeros, the largest number of zeros in a row. For instance, the longest run of zeros in [1,0,1,1,0,0,0,0,1,0,0] is 4.**

Program:

```
import random
c=max_c=0
l=[] for i in range(1,101):
x=random.randint(0,1)
   l.append(x)
print(l)
for j in l:
   if j==0:
    c+=1 else:
       if c>max_c:
          max_c=c
       c=0
print("longest run of zeros=",max_c)
```

Q15)**Write a program that removes any repeated items from a list so that each item appears at most once. For instance, the list [1,1,2,3,4,3,0,0] would become [1,2,3,4,0]. program:**

```
l=[1,1,2,3,4,3,0,0]
li=[] for
i in l:
    if i not in li:
        li.append(i)
print(li)
```

#17) Write a function called sum_digitsthat is given an integer num and returns the sum of the digits of num.

```
def sum_digit(num):
    sum=0
    while(num>0):
    x=num%10 sum+=x
    num=num//10
    return sum
print(sum_digit(153))
```

#18) Write a function called first_diff that is given two strings and returns the first location in which the strings differ. If the strings are identical, it should return -1

```
def fun(str1,str2):
    if str1==str2:
        print("Both the strings are Identical",-1)
    else:
        if len(str1)>len(str2): for i
            in range(len(str2)):
                if str1[i]!=str2[i]:
                    print(str1.index(str1[i]))
                    break
              else:
                print(i+1)
        elif len(str2)>=len(str1):
            for i in range(len(str1)):
                if str1[i]!=str2[i]:
                    print(str2.index(str2[i]))
                    break
            else:
            print(i+1)

str1=input("enter string1=")
str2=input("enter string2=")
fun(str1,str2)
```

#19)Write a function called number_of_factorsthat takes an integer and returns how many factors the number has.

```
def num_fact(num):
    factorspos = []
    print("The factors of",num,"are:",end='')
    if num > 0: # if input is postive for i in
    range(1,num+1): if num % i == 0:
            factorspos.append(i)
        print(factorspos,'total number of factors=',len(factorspos))
    elif num < 0: #if input is negative
        for i in range(num,0):
            if num % i == 0:
                factorspos.append(i)
        print(factorspos,'total number of factors=',len(factorspos))
    else:
        print('Not Possible')
num=int(input('Enter  a number value='))
num_fact(num)
```

#20)Write a function called is_sortedthat is given a list and returns True if the list is sorted and False otherwise def is_sorted(string): l=list(string) if l==sorted(l):

```
        print('True')
    else:
        print('False')

string=input('enter string ')
is_sorted(string)
```

(OR)

```
def is_sorted(l):
    k=l[:]
    k.sort()     print("The K List
elements are",k)     if(l==k):
        return  True
else:          return
False
```

#21)Write a function called root that is given a number x and an integer n and returns x1/n. In the function definition, set the default value of n to 2.

```
def root(num):
    return (num**0.5)
num=int(input('enter number ='))
root(num)
```

#22)Write a function called primes that is given a number n and returns a list of the first n primes. Let the default value of n be 100 def printPrime(n=100):

```
    l=[] x=2
    while(len(l)<n): for i
    in range(2,x):
            if(x%i==0):
                break
        else:
            l.append(x)
        x=x+1
    return(l)
n=int(input("enter the number of prime numbers:"))
s=printPrime(n)
print("List of first",n,"primes:",s)
j=printPrime()
print("List of first 100 primes:",j)
```

#23 Write a function called merge that takes two already sorted lists of possibly different lengths, and merges them into a single sorted list.

```
def merge(x,y):
    X=sorted(x)
    Y=sorted(y)
    print(X) print(Y)
    print(sorted(X+y)
    )
def merge1(x,y):
    x.extend(y) for i in
    range(0,len(x)):
        for j in range(0,len(x)):
            if x[i]<x[j]:
                t=x[i]
                x[i]=x[j]
                x[j]=t print(x)
x=[2,4,3,1]
y=[0,6,5,8,7,9]
merge(x,y)
merge1(x,y)
```

24) Write a program that asks the user for a word and finds all the smaller words that can be made from the letters of that word. The number of occurrences of a letter in a smaller word can't exceed the number of occurrences of the letter in the user's word.

```
from itertools import permutations

w=input("Enter  A Word:") for i in

range(2,len(w)):            for   p   in
```

```
permutations(w,i):

print(''.join(p),end=' ')
```

25) Write a program that reads a file consisting of email addresses, each on its own line. Your program should print out a string consisting of those email addresses separated by semicolons.

```
file=open(input("enter file name"),"r")
lines=file.readlines() print(lines) for
line in range(len(lines)):
   if(line==len(lines)-1):
      print('{}'.format(lines[line].strip()))
   else: print('{}'.format(lines[line].strip()),
      end=";")
```

26) Write a program that reads a list of temperatures from a file called temps.txt, converts those temperatures to Fahrenheit, and writes the results to a file called ftemps.txt.

```
file=open('temp.txt','r')

l=file.readlines()

f=open('ftemps.txt','w')

for line in range(len(l)):

   d=l[line].strip()
   fahrenheit = (float(d) * 9/5) + 32

   f.write(str(fahrenheit)+"\n")

f.close()
```

27) Write a class called Product. The class should have fields called name, amount, and price, holding the product's name, the number of items of that product in stock, and the regular price of the product. There should be a method get_price that receives the number of items to be bought and returns the cost of buying that many items, where the regular price is charged for orders of less than 10 items, a 10% discount is applied for orders of between 10 and 99 items, and a 20% discount is applied for orders of 100 or more items. There should also be a method called make_purchase that receives the number of items to be bought and decreases amount by that much.

```
class product:
    def __init__(self,name,items,price):
        self.name=name
        self.items=items
        self.price=price
    def getprice(self,n):
        if n<10:
            print("regular price is charged for your
            order") cost=n*self.price print("actual
            cost=",cost)
        elif n>=10 and n<100: print("10% discount is
            applied for your order") cost=n*self.price
            discount=(cost*10)/100 finalcost=cost-
            discount print("actual cost=",cost) print("10%
            discount=",finalcost) print("cost after 10%
            discount=",discount)
        else:
            print("20% discount is applied for your order")
            cost=n*self.price discount=(cost*20)/100
            finalcost=cost-discount print("actual
            cost=",cost) print("20% discount=",finalcost)
            print("cost after 20% discount=",discount)
    def my_purchase(self,n):
        if n<10:
            print("regular price is charged for your
            order") cost=n*self.price print("final
            cost=",cost)
        elif n>=10 and n<100: print("10% discount is
        applied for your order") cost=n*self.price
        discount=(cost*10)/100 finalcost=cost-discount
        print("actual cost=",cost) print("10%
        discount=",finalcost) print("cost after 10%
        discount=",discount) else:
            print("20% discount is applied for your order")
            cost=n*self.price discount=(cost*20)/100
            finalcost=cost-discount print("actual
            cost=",cost) print("20% discount=",finalcost)
            print("cost after 20% discount=",discount)
```

```
p=product("pen",200,5)
n=int(input("enter number of pens you want to buy"))
p.getprice(n)
n=int(input("enter number of pens you want to buy"))
p.my_purchase(n)
```

28)Write a class called Time whose only field is a time in seconds. It should have a method called convert_to_minutes that returns a string of minutes and seconds formatted as in the following example: if seconds is 230, the method should return '5:50'. It should also have a method called convert_to_hours that returns a string of hours, minutes, and seconds formatted analogously to the previous method.

```
class Time:     def __init__(self,sec):
      self.sec=sec     def
seconds_to_Minutes(self):
         s=self.sec           minutes = s // 60           seconds=s%60
print('Times in Minutes:seconds format',str(minutes)+":",str(seconds))     def
seconds_to_Hours(self):
         s=self.sec           hours = s // 3600           minutes=(s//60)%60           seconds=s%60
print('Times in Hours:Minutes:seconds format',str(hours)+":",str(minutes)+":",str(seconds))
sec=int(input('Enter the number of seconds:'))
obj=Time(sec)
obj.seconds_to_Minutes()
obj.seconds_to_Hours()
```

30) Write a Python class to implement pow(x, n).

```
class Pow1:
   def poe(self,x,n):
      k=x**n
print("pow({},{})={}".format(x,n,k))

p=Pow1()
x=int(input("Enter the X value:"))
n=int(input("Enter the Y value:"))
p.poe(x,n)
```

31) Write a Python class to reverse a string word by word.

```
class Sa:

    def display(self,s):          words = s.split(' ')

reverse_sentence = ' '.join(reversed(words))

print("The reverse words of string:",reverse_sentence)


k=Sa()

s = input("enter the string:")

k.display(s)
```

32) Write a program that opens a file dialog that allows you to select a text file. The program then displays the contents of the file in a textbox.

Case study of file systems

```
COMMANDS = ('1', '2', '3', '4', '5', '6', '7')

MENU = """1 List the current directory

2 Move up

3 Move down

4 Number of files in the directory

5 Size of the directory in bytes

6 Search for a filename

7 Quit the program"""

QUIT='7'

import os

def main():
    while True:
        print(os.getcwd())
        print(MENU)
        command=acceptcommand()
        runcommand(command)
        if command==QUIT:
            print('thank you')
            break
def acceptcommand():
    '''inputs and return command number'''
    command=input('enter a number')
    if command in COMMANDS:
        return command
    else:
        print('error :command not recognized')
        return acceptcommand()
def runcommand(command):
    if command=='1':
        listCurrentDir(os.getcwd())
```

```
    elif command=='2':
        moveUp()
    elif command=='3':
        moveDown(os.getcwd())
    elif command=='4':
        print('the total number of files=',countFiles(os.getcwd()))



    elif command=='5':
        print('total number of bytes',countBytes(os.getcwd()))
    elif command=='6':
        findFiles()

def listCurrentDir(dirname):
    '''print list of the cwd contents'''
    lyst=os.listdir(dirname)
    for i in lyst:print(i)
def moveUp():
    '''moves up the parent directory'''
    os.chdir('..')
def moveDown(currentDir):
    '''moves down to the named subdirectory if it exists'''
    newdir=input('enter the directory name')
    if os.path.exists(newdir):
        os.chdir(newdir)
    else:
        print('no such name')

def countFiles(path):
    """Returns the number of files in the cwd and
      all its subdirectories."""
```

```
    count = 0
    lyst = os.listdir(path)
    for element in lyst:
        if os.path.isfile(element):
          count += 1
        else:
          os.chdir(element)
          count += countFiles(os.getcwd())
          os.chdir("..")
    return count
def countBytes(path):
    newdir=input('enter dir name')
    print(os.path.getsize(newdir))
def findFiles():
    newfile=input('enter file name')
    if os.path.isfile(newfile):
        print('exist')
    else:
        print('no such file in current path')
main()
```

33) Write a program to demonstrate Try/except/else.try:

```
  a=int(input("enter the A value:"))
b=int(input("enter the B value:"))
s=a/b except ZeroDivisionError:
   print("Can't divide by zero(b=0)")
else:    print("The division is=",s)
```

34) Write a program to demonstrate try/finally and with/as.

```
    try:
          a=int(input("enter the A value:"))
        b=int(input("enter the B value:"))
        s=a/b except ZeroDivisionError:
           print("Can't divide by zero(b=0)") else:
           print("The divsion is=",s) finally:
           print('This is always executed')
```