

Program 21

Aim: Count the number of character (character frequency) in a string.

Program

```
test_str = "hai ashique how are you"  
count = 0  
for i in test_str:  
    if i == 'a':  
        count = count + 1  
print ("count of a string is :" + str(count))
```

Result

The program has been executed and output has been verified.

of a morph

Output

read more upto others becoming part of edgeC and
count of a string is: 3

A S
P S C
S1 S1 S A

import

(("read more upto others") form) for <0
I = 9

: (read) spear di i rot

I = 0

: (more) spear di C rot

I = 1 S1 = spear

((" + base, spear) form)

(" ") form

free

base, speakers and read, comprising set

base is now higher

Program 22

Aim: Adding at the end of a given string, if it already ends with ing; then add ly??
E.g. for a string for example: str1 = "string".
program

```
def add_string(str1):
    length = len(str1)
    if length > 2:
        if str1[-3] == 'ing':
            str1 += 'ly'
        else:
            str1 += 'ing'
    return str1
print(add_string('as'))
print(add_string('ash'))
print(add_string('string'))
```

Result

The program has been executed and output was verified.

16 corporal

Output
returning the words to reshape left-hand side
as points on C (corporal)

as

existing

Stringly, paper work problems instead of test

o - forms

intestine or I not

IP = 34,

1 + forms = forms

((hand) etc + "or points to forms") form

Hung

1 fugitive has broken and will corrupt self

bodily and

Program 23

Aim: Accept a list of words and return length of longest word.

Program

```
a = []
n = int(input("Enter the number of element in list : "))
for x in range(0, n):
    element = input("Enter element " + str(x+1) + ": ")
    a.append(element)
max1 = len(a[0])
temp = a[0]
for i in a:
    if (len(i) > max1):
        max1 = len(i)
        temp = i
print("The word with longest length is : ")
print(temp)
```

Result

The program has been executed and output is verified.

program

Output

Enter the number of elements in list : 3

Enter element 1: Apple

Enter element 2: Banana

Enter element 3: Strawberry

The word with the longest length is : Strawberry

Program 24

Output

Aim: Construct following pattern using nested loop?

```
*  
* *  
* * *  
* * * *  
* * * * *  
* * * *  
* * *  
* *  
*
```

```
1  
2  
3  
4  
5  
01  
02  
03  
04  
05  
06  
07  
08
```

Program

```
n=5  
for i in range(n):  
    for j in range(i):  
        print('*', end=' ')  
    print()  
for i in range(n, 0, -1):  
    for j in range(i):  
        print('*', end=' ')  
    print()
```

Result

The program has been executed and output was verified.

B6 compact

Output

to offport center bars throw to tail or high A mark
brace forward

*

* *

* * *

* * * *

* * * * *

* * * * *

* * * *

* * *

*

compact

[J=0]

((*: hit at "lowest to radarsoft ratio") fudge) tail-a

: (0,0) appear in L inf

((*: hit at "lowest ratio") fudge) = forward

: (lowest) begin. B

([0] B) inf = [700]

[0] B = quat

: B on rot

: (180m < (i) m1) +

(i) m1 = 180m

i = quat

((*: offport fudge after brace soft) fudge

(quat) fudge

first

fudge has biting mid and compact soft

bite

Program 25

Aim: Generate all factors of number or a number?

Program

```
def print_factors(x):
    print('The factors of ', x, 'are :')
    for i in range(1, x+1):
        if x % i == 0:
            print(i)
num = 200
print_factors(num)
```

Result

The program has been executed and output was verified.

Output

No concept

The factors of 200 are: general fraction not

1
2
4
5
8
10
20
25
40
50
100
200

1
2
4
5
8
10
20
25
40
50
100
200

concept

: (i) open circuit

: (ii) open circuit

(i + 100, (i + 2) loop

(i) loop

: (i + 100) open circuit

: (ii) open circuit

(i + 100, (i + 2) loop

(i) loop

open circuit loops between coil and coupling coil

Program Q6

Aim: Use lambda functions to find area of square, rectangle & triangle

Program

[aa,ee,sh,pe,bb,ii]

```
import math
t-peri = lambda p,q,r : p+q+r
r-area = lambda len, ht : len * ht
c-peri = lambda rad : 2 * math.pi * rad
c-area = lambda rad : math.pi * rad * rad
print ("perimeter of triangle(10,20,15) is:", t-peri(5,20,10))
print ("Area of rectangle (30,20) is:", r-area (20,10))
```

Result

[aa,ee,ii]

The program has been executed and output was verified.

Output

! continue to calculate for next. No informed result

Perimeter of triangle (10,20,15) is : 35

Area of rectangle (30,20) is : 200

(*) calculate long fib
(*) simple fib without sqrt
(*) square root not
 ! 0 = fib 1
 (*) fib
 0 or 1 = fib
(conv) calculate fib

cross fib two times and add overlapping part
 ! fib 100

Program 27

Aim: form a list of integers, create a list removing even numbers

Program

```
list = [11, 29, 33, 44, 55, 66]
print("original list")
print(list)
for i in list:
    if (i % 2 == 0):
        list.remove(i)
print("list after removing an even number:")
print(list)
```

Result

The program has been executed and output was verified.

Q6 Output

Output

Output: [11, 22, 33, 44, 55, 66]

Original list

[11, 22, 33, 44, 55, 66]

list after removing an even numbers

[11, 33, 44, 55, 66]

list after removing an even number

[11, 33, 55, 66]

list after removing an even number

[11, 33, 55]

also function has behavior and will drop off

Program 28

Aim: Display future leap years from current year to a final year entered by the user.

[Ans , P6.3 -

Program

```
import datetime  
a = datetime.datetime.now()  
a = int(a.year)  
b = int(input('Enter final year:'))  
print("In leap years")  
for i in range(a, b+1):  
    if (i%4 == 0):  
        print(i)
```

Result

The program has been executed and output verified.

Output

* Enter final year: 2050

leap years:

2024

2028

2032

2036

2040

2044

2048

Program 29

Aim: Generate positive list of numbers from a given list of integers.

Program

```
list1 = [5, -8, 69, 57, -55, 24, -66, -20, 80, -33, -639, 852]
pos = list()
for i in list1:
    if i > 0:
        pos.append(i)
print('original list:', list1)
print('positive integer list:', pos)
```

Result

The program has been executed and output verified.

88. Output

Output

Print range function and print good method value : min
Original list : [5, -8, 69, 57, -55, 24, -66, -20, 80, -33,
-639, 852]

Positive integers list : [5, 69, 57, 24, 80, 852]

(i) `map(lambda x: x if x > 0 else 0, list)`
(ii) `filter(lambda x: x > 0, list)`
(iii) `list(filter(lambda x: x > 0, list))`
(iv) `list(map(lambda x: x if x > 0 else 0, list))`

Ans

shadow fugitive has broken and will continue soft

Program 30

Aim: find biggest of 3 numbers entered

program

```
a = int(input('Enter 1st no: '))
b = int(input('Enter 2nd no: '))
c = int(input('Enter 3rd no: '))
if a > b and a > c
    print(a, 'is the biggest number')
elif b > a and b > c
    print(b, 'is the biggest number')
else:
    print(c, 'is the biggest number')
```

Result

The program has been executed and verified.

Program

Output

Enter 1st no: 562

Enter 2nd no: 960

Enter 3rd no: 750

960 is the biggest number

Program 31

Aim: Create a list of colors from comma-separated color names entered by user. Display first and last colors.

Program

```
colors = input('enter colors separated by commas:'),  
           split(',')  
print('first color: ', colors[0])  
print('last color: ', colors[-1])
```

Result

The program has been executed and output verified.

35 complete

Output

border pattern & background file name

Enter colors separated by commas: blue, green,
red, black, yellow

first color: blue

((1:255,0,0)) foreground

last color: yellow

((0,255,255,0)) background

3rd border color

((red,blue,green,black,white)) temp

3rd border color file

((red,blue,green,black,white)) border

3rd

((red,blue,green,black,white)) temp

different from below and will merge with

Program 32

Aim: Print out all colors from color-list 1 not contained in color-list 2.

Program

```
colors 1 = Set((input('Colour separated by commas:')).split(','))  
colors 2 = Set((input('Colour separated by commas:')).split(','))  
print ('Colors in color-list 1 not contained in color-list 2  
are: ', list(colors 1.difference(colors 2)))
```

Result

The program has been executed and output is verified.

16. *intersection*

Output

↳ *intersection* method generates list of shared colors between two input polygons. *newest* method creates intersection region.

Color separated by commas : red, orange, green, black

Color separated by commas : blue, yellow, gold, brown }

Color in color-list1 not contained in color-list2 are :

↳ *color-list1* [red, orange, green, black] *color-list2* [blue, yellow, gold, brown]

(0) *color-list1* [red, orange, green, black] *color-list2* [blue, yellow, gold, brown]

Illustration

↳ *intersection* function, *newest* method and *intersection* method

Program 33

Aim: Create a single string separated with space from two strings by swapping the character at position 1.

Program

```
Str 1 = Input ('Enter a string : ')
Str 2 = input ('Enter another string : ')
Str 3 = Str2 [0] + Str 1 [1] + " " + Str1 [0] + Str 2 [1]
print (Str 3)
```

Result

The program has been executed and output verified.

Q3 (Method)

Output

Enter a string : ashique

Enter another string: kozhikode

kshique aozhikode

ashique kozhikode

kozhikode kshique

kozhikode aozhikode

function & loops function mod and compare with

Program 34

Aim: Create a package graphics with modulus rectangle, circle and sub package 3D-graphics with modulus cuboid and sphere include method to find area and perimeter of respective figure in each module. Write programs that finds area and perimeter of figure by different importing statements.
(include selective import of modules and import * statements)

Program

→ circle.py

findarea.py

```
import circle
from rectangle import *
from Graphics_3D_graphics import cuboid, sphere
a = float(input('Enter length of the rectangle: '))
b = float(input('Enter breadth of the rectangle: '))
area(a,b)
r = float(input('Enter the radius of circle: '))
circle.area(r)
l = float(input('Enter length of the cuboid: '))
b = float(input('Enter breadth of the cuboid: '))
h = float(input('Enter height of the cuboid: '))
cuboid.area(l,b,h)
r = float(input('Enter the radius of the sphere: '))
sphere.area(r)
```

→ findperimeter.py

Import circle

from rectangle import *

from graphics_3D-graphics import cuboid, sphere

a = float(input('Enter length of the rectangle:'))

b = float(input('Enter breadth of the rectangle:'))

rectangle.perimeter(a,b)

r = float(input('Enter the radius of the circle:'))

circle.circumference(r)

l = float(input('Enter length of the cuboid:'))

b = float(input('Enter breadth of the cuboid:'))

h = float(input('Enter height of the cuboid:'))

cuboid.perimeter(l,b,h)

s = float(input('Enter the radius of the sphere:'))

sphere.perimeter(s)

→ rectangle.py

def area(a,b):

print('Area of rectangle with sides ', 'a', 'and ', 'b', 'is', ;
 % .2f '% (a*b), 'sq.units')

def perimeter(a,b)

print('Perimeter of rectangle with sides ', 'a', 'and ', 'b', 'is', ;
 ' % .2f '% (2 * (a+b)), 'units')

→ 3D-graphics

cuboid.py

def area(l,b,h):

print('Total surface area of cuboid with dimensions',
 'l', 'b', 'h', 'is', '% .2f', '% .2f' ((l*b)+ (b*h)+
 (l*h))), 'squnits')

```
def perimeter(l,b,h):
```

```
    print('perimeter of cuboid with dimensions', 'l', 'b', 'h'  
         'is', '%', '2f' % (4 * (l+b+h)), 'unit')
```

→ Sphere.py

```
def area(r):
```

```
    print('Area of sphere with radius', 'r', 'is', '  
          %2f' % (3.14 * r * r)), 'sq. units')
```

```
def perimeter(r):
```

```
    print('perimeter of (great circle of) sphere with  
radius', 'r', 'is', '%2f' % (2 * 3.14 * r), 'units')
```

Result

The program has been executed and output is verified.

Output

- Perimeter of circle with radius 10 is 62.83185307179586
- Area of a circle with radius 10 is 314.1592653589793
- Area of triangle with length and width 10 is 100
- Perimeter of a rectangle with length and width 10 is 40
- Area of cuboid with length, width, height 10 is 8600
- Perimeter of cuboid with length, width, height 10 is 120
- Area of sphere with radius 10 is 12566370614359173
- Perimeter of sphere with radius 10 is 62.8318530717995

Program 35

Aim: Create Rectangle class with attributes length, breadth and methods to find area and perimeter. Compare two rectangle objects by their area.

Program

class Rectangle:

def __init__(self, l, b):

self.length = l

self.breadth = b

def area(self):

return self.length * self.breadth

def perimeter(self):

return self.length + self.breadth

def cmp(self, obj):

if self.area() > obj.area():

print('Rectangle with length = ', self.length,
and breadth = ', has the greater area')

elif self.area() > obj.area():

print('Rectangle with length = ', obj.length,
and breadth = ', object.breadth, ' has
the greater area')

else:

print("They have equal area")

r1 = Rectangle(6, 5)

r2 = Rectangle(8, 4)

r1.cmp(r2).

Result

The program has been executed and output is verified.

Output

Rectangle with length = 8 and breadth = 4 has the greater area.

Program 36

Aim: Create a Bank account with members account number, name, type of account and balance. Write constructor and methods to deposit at the bank and withdraw an amount from the bank.

Program

Class Bank Account:

```
def __init__(self, a, t, b):
```

```
    self.acno = a
```

```
    self.name = n
```

```
    self.type = t
```

```
    self.bal = b
```

```
def deposit(self, a):
```

```
    self.bal += a
```

```
    print("Rs", a, "deposited! current balance is:
```

```
    Rs", self.bal)
```

```
def withdraw(self, a):
```

```
    if self.bal >= a
```

```
        self.bal -= a
```

```
    print("Rs", a, "withdrawn! current
```

```
balance is: Rs", self.bal)
```

```
else:
```

```
    print("Insufficient balance to make this
```

```
transaction!")
```

```
a = int(input("Enter Account number:"))
```

```
n = input ("Enter name of the account holder :")  
t = input ("Enter Account type : ")  
b = float (input ("Enter your balance : "))  
ad = BankAccount (a,n,t,b)  
act1.deposit (float (input ("Enter amount to deposit : ")))  
act1.withdraw (float (input ("Enter amount to withdraw ")))
```

Result

The program has been executed and output is verified.

Output

Enter account number : 110110112135

Enter name of the account holder : Ashique

Enter account type : zero balance account

Enter your balance : 10000

Enter amount to deposit : 1000

Rs 1000.0 deposited! Current balance is Rs. 11000.0

Enter amount to withdraw : 500

Rs 200.0 is withdrawn! current balance is Rs 10800.0

Program 37

Aim: Create a class rectangle with private attributes length & width. Overload '`<`' operator to compare the area of 2 rectangle.

Program

class Rectangle:

def __init__(self, l, w):

self.length = l

self.width = w

self.area = self.width * self.length

def __lt__(self, other):

if self.area < other.area:

print("Rectangle with length = " + str(self.length) +
"; and width = " + str(self.width) + ", has
the lesser area !")

elif

other.area > self.area:

print("Rectangle with length = " + str(other.length) + " and
width = " + str(other.width) + " has the lesser area")

else

print("They have equal area !")

`l = float(input("Enter length of 1st rectangle:"))`

`w = float(input("Enter width of 1st rectangle:"))`

`R1 = Rectangle(l, w)`

```
l = float (input ("Enter length of 2nd rectangle :"))
```

```
w = float (input ("Enter lengthwidth of 2nd rectangle :"))
```

```
R2 = Rectangle (l,w)
```

```
R1 < R2
```

Result

The program has been executed and output is verified.

Output

Enter length of 1st rectangle: 5

Enter width of 1st rectangle: 3

Enter length of 2nd rectangle: 9

Enter length of 2nd rectangle: 6

Rectangle with length=5 and width=3 has the
area.

Program 38

Aim: Create a class Time with private attribute hour, minutes and seconds. Overload + operator to find sum of 2 time.

Program

Class Time:

```
def __init__(self, hh=0, mm=0, ss=0):
    self.__hour = hh
    self.__second = ss

def __add__(self, other):
    second = int((self.__second + other.__second) % 60)
    minute = int((self.__minute + other.__minute) % 60 +
                  (self.__second + other.__second) // 60))
    hour = int((self.__hour + other.__hour) * 60 * 60 +
               (self.__minute + other.__minute) // 60)

    print(f'{time[hh:mm:ss]}; {hour}:{minute}:{second}')
```

$T_1 = \text{Time}(2, 25, 15)$
 $T_2 = \text{Time}(18, 50, 45)$
 $T_1 + T_2$

Result

The program has been executed and output is verified.

Output

Time [hh:mm:ss] 21:16:30