

26/2/2020

Programs:

Aim :- Python programs to find area of a circle ($A = \pi r^2$) [using functions]

Procedure :- # Area of a Circle

From math import pi

```
r = float(input("Enter the radius of the circle:"))
print("the area of the circle with radius" +
      str(r) + " is :" + str(pi * r * r))
```

Result : the program has been executed and the output was verified.

Output

Enter the radius of the circle : 4
the area of the circle radius 4 is 50.2654

28/2/21

Programs : 2

Aims: Python program to find largest among three numbers.

Procedure :-

```
# Largest among three numbers.  
num1 = float(input("Enter first number"))  
num2 = float(input("Enter second number"))  
num3 = float(input("Enter third number"))  
if (num1 > num2) and (num1 > num3): largest = num1  
elif (num2 > num1) and (num2 > num3): largest = num2.  
else: largest = num3  
print("The largest number is ", largest)
```

Result: the program has been executed and the output was verified.

3rd program

problem :- find maximum of three numbers with condition

Output :- 33.0

Enter first number : 33

Enter second number : 25

Enter third number : 44

the largest number is 33.0

3rd program :- find maximum of three numbers with condition

Program - 3

Aim:- Python program to find square of a number

Procedure:

```
# Square of a number  
for x in range(100):  
    print(x*x)  
digit = int(input("Enter an integer number:"))  
Square = digit * digit  
print(f'Square of {digit} is {Square}')
```

Result: the program has been executed and the output was verified.

Output

Output
Number and square equal.

Enter an integer number : 4

Square of 4 is 16

Number and square equal, both are equal

Number and square both are equal to 16

Program 4

Aim :- Python programs to find square of n numbers in list.

```
[list = [12, 13, 14, 15]]
```

```
for n in list:
```

```
    print(n*n)
```

Procedure :-

```
# Square of n number in list
```

```
list = [12, 13, 14, 15]
```

```
for n in list:
```

```
    square = n*n
```

```
    print(n, square, ":", "square")
```

Result : the program has been executed and
the output was verified.

Output

12 Square is : 144

13 Square is : 169

14 Square is : 196

15 Square is : 225

Program : 5

Aim: Python program to find vowels in a string

Procedure:

```
# vowels in a string
```

String A = "Hello ... how are you"

```
Print (" Given String :\n", StringA)
```

Vowels = "AaEeIiOoUu"

```
res = Set ([each for each in StringA if each  
in Vowels])
```

```
Print (" The vowels present in the String :\n", res)
```

Result: the program has been executed and the output was verified.

Program

Find out number of vowel & consonant in a string

Output: 4, 6

Given String:

("Hello")

Hello... how are you

The vowels present in the string:

{'a', 'o', 'u', 'e'}

Counting the vowels in a string

Program 6

Also: Python program to count words in a sentence

Procedure :-

Count words in a Sentence

Original_string = "Python is an interpreted high-level
programming and general-purpose programming
language"

print("The Original String is :" + original_string)

result = len(Original_string.split())

print ("The number of words :" + str(result))

Result : The program has been executed and the output was
verified.

part 2 of answer book of computer architecture lesson 4

Output

Original string = 'Python is an interpreted high-level programming and general-purpose programming language'

The number of words : 12

"Hello! I'm Python" - answer

Ans 4) A part of answer is (12) + 2 = 23

(answer a)

(12) + 2 = 14

Answers with two blanks and not compare with final answer now

Programs: 7

Aim:- Python program to count a letter 'a' in a list.

Procedure:-

```
# count a letter a in a list
```

```
a = ['Aiswarya', 'Amala', 'Sujith', 'Sumith']
```

```
str1 = ('.' join (a))
```

```
Count = 0
```

```
For i in str1:
```

```
If i == 'a':
```

```
Count = Count + 1
```

```
Print ("count of 'a' in the list is: " str (Count))
```

working in all classes takes of comparing methods with

Output

Count of 'a' in the list is : 6

Scoping down to the parameters

(parameters)

(parts) Inargs + * args keepers with ") part

(args parts keepers) and = Max

((max) else it's shown as minimum with () part)

new things will too believe and not compare with Max
maximum

Program : 8

Aim :- Python program to check the length of a list.

Procedure :

```
# check the length of a list
```

```
list1 = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110]
```

```
list2 = [140, 120, 130, 140, 150, 160, 170, 180, 190, 200, 210]
```

```
len1 = len(list1)
```

```
len2 = len(list2)
```

```
if len1 == len2:
```

```
    print('both list have a equal length')
```

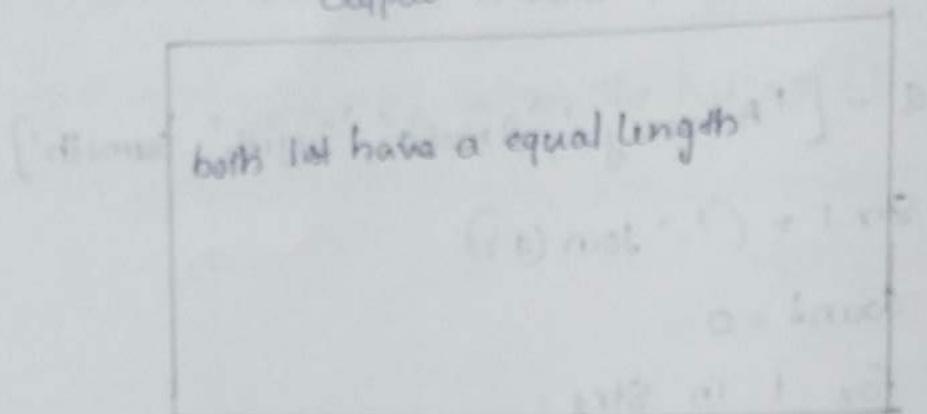
```
else:
```

```
    print('both list does not have equal length')
```

Result : The program has been executed and the output was verified.

make a new list by taking elements from both lists sequentially

(ii) Output will be having all



if both list are empty

then output is empty

knowing both list will not go beyond their length

Program 9

Aim:- Python program to check the sum of list

Procedure:-

Sum of list

total = 0

List1 = [11, 5, 17, 18, 23, 25, 70, 31]

for ele in range(0, len(List1)):

 total = total + List1[ele]

print("Sum of all elements in given list:", total)

Result: the program has been executed and the output was verified.

Output - print all sum = 16

Sum of all element in given list : 200

Aims :- Python program to check the common elements in the list.

Procedure:-

Common elements in the list.

list1 = [12, 16, 15, 16, 14, 13, 12, 12, 11, 10, 10]

list2 = [12, 10, 12, 11, 10, 16, 15, 14, 13, 12, 16]

for value in list1:

 if value in list2:

 common = 1

 if common == 1:

 print('there are common elements')

else:

 print('no common elements')

Program 1

and it must return at least one output - which

Output

there are common elements

O = 1st

[10, 20, 30, 40, 50, 60] = 100

(10) 10 appears on the set

[10] 10 is a value - int

Output is the output of elements in go much faster.

Let's see how iteration and outputting will change

Iteration now

Program: 11

Aim: Python program to replace a character in a string.

Procedure:-

Python program to replace a character in a string

str1 = input("Please Enter your own string : ")

ch = input("Please Enter your own character : ")

newch = input("Please Enter the new character : ")

str2 = ''

for i in str1:

if (i == ch):

str2 = str2 + newch

else:

str2 = str2 + i

print("Original String : ", str1)

print("Modified String : ", str2)

Result: the program has been executed and the output was verified.

Output

Please Enter your own string : Swothy

Please Enter your own character : 4

Please Enter your own the new character : i

Original String : Swothy

Modified string : Swothiy

Aim :- Python Program to exchange the first and Last letter in a string.

Procedure :-

```
# Exchange the first and Last letters in a string  
def exchange(str)  
    return str[-1] + str[1:-1] + str[0]  
print(exchange('rainbow'))
```

Result: The program has been executed and output was verified.

10/10/19

Grade is an indicator of multiple of anything creditable

Grade is an indicator of multiple of anything creditable

Output of output test

wainbow

rainbow with length = 1.0

rainbow with length = 0.5

rainbow with length = 0.25

rainbow with length = 0.125

rainbow with length = 0.0625

rainbow with length = 0.03125

rainbow with length = 0.015625

rainbow with length = 0.0078125

rainbow with length = 0.00390625

rainbow with length = 0.001953125

rainbow with length = 0.0009765625

rainbow with length = 0.00048828125

rainbow with length = 0.000244140625

rainbow with length = 0.0001220703125

rainbow with length = 0.00006103515625

rainbow with length = 0.000030517578125

rainbow with length = 0.0000152587890625

rainbow with length = 0.00000762939453125

rainbow with length = 0.000003814697265625

rainbow with length = 0.0000019073486328125

rainbow with length = 0.00000095367431640625

rainbow with length = 0.000000476837158203125

rainbow with length = 0.0000002384185791015625

rainbow with length = 0.00000012020928955078125

rainbow with length = 0.000000060104644775390625

rainbow with length = 0.0000000300523223876953125

rainbow with length = 0.00000001502616119384765625

rainbow with length = 0.000000007513080596923828125

rainbow with length = 0.0000000037565402984619140625

rainbow with length = 0.00000000187827014923095703125

rainbow with length = 0.000000000939135074615478515625

rainbow with length = 0.0000000004695675373077392578125

rainbow with length = 0.00000000023478376865386962890625

rainbow with length = 0.000000000117391884326934814453125

rainbow with length = 0.0000000000586959421634674072265625

ratio of two identical and equal charges with largest
length ratio

Aim :- Python program to merge two dictionaries

Procedure :-

```
# merge two dictionaries
```

```
ds = {'a':100,'b':200}
```

```
ds = {'c':300,'d':400}
```

```
dictionary = ds.copy()
```

```
dictionary.update(ds)
```

```
print("the merged dictionary is:", dictionary)
```

Result: the program has been executed and output was verified.

Method to work with operations of compound objects - : with
. periods & as well

Output

the merged dictionary is $\{a':100, b':200, c':300, d':400\}$

(a) operator $+$

(d) $a + b + c + d$ $\Rightarrow \{100, 200, 300, 400\}$

(C) operator $+$ list

Aim:- Python program to ascend and descent dictionary.

Procedure:-

* ascend and descent dictionary

import operator

d = {1:2, 3:4, 4:3, 2:1, 0:0}

print ("Original dictionary : ", d)

Sorted_d = sorted ((d.items(), key = operator.itemgetter(1)))

print ("Dictionary in ascending order by value : ",
Sorted_d)

Result: The program has been executed and output was verified.

Output

Original dictionary: $\{1:2, 3:4, 4:3, 2:1, 0:0\}$

Dictionary in ascending order by value:

$[(0,0), (2,1), (1,2), (4,3), (3,4)]$

Dictionary in descending order by value:

$\{3:4, 4:3, 1:2, 2:1, 0:0\}$

9/3/21

Programs: 15

Aim:- Python program to find remove even numbers from a list.

Procedure :-

remove even numbers from a list

List = [1, 2, 4, 5, 6, 8, 10, 11, 13]

for i in list - copy ():

divid = i % 2

If divid == 0:

list.remove(i)

print(list)

Result: The program has been executed and output was verified.

output

[1, 5, 11, 13]

{0, 1, 2, 3, 4, 5, 6, 7, 8}

5/3/21

Programs: 16

Aim :- Python program to find gcd of a number.

Procedure:-

Find gcd of a number

num1 = int (input ("Enter a 1st number:"))

num2 = int (input ("Enter the 2nd number:"))

i=1

while i <= num1 and i <= num2:

if (num1 % i == 0 and num2 % i == 0):

gcd = i

i = i+1

print ("gcd is", gcd)

Result: the program has been executed and output was verified.

Output

```
Enter a start number : 5  
Enter the end number : 15  
gcd is 5
```

Aims:- Python Program to Find Factorial up a number.

Procedure :-

```
# Find Factorial of a number
# Factorial number provided by a user

num = 7 # change value for different result
# num = int(input("Enter a number :"))

factorial = 1
if num < 0:
    print("Sorry, factorial does not exist for negative numbers")
elif num == 0:
    print("The Factorial of 0 is 1")
else:
    for i in range(1, num+1):
        factorial *= i
    print("The Factorial of", num, "is", factorial)
```

Result: the program has been executed and output was verified.

at program

Materials & following part of program code will work

for the output

Output

The Factorial of 7 is 5040.

i = 5040

i = 5040

(last, 5040) found

After few iterations, code will converge with desired value.

Aim:- Python program to find Fibonacci series.

Procedure:-

Python program to generate Fibonacci Series

n = input("Enter the value of 'n': ")

a = 0

b = 1

Sum = 0

Count = 1

print("Fibonacci Series:", end = " ")

while (Count <= n):

print(Sum, end = " ")

Count += 1

a = b

b = Sum

Sum = a + b

Result : the program has been executed and output was verified.

Output

Enter the value of 'n': 5
Fibonacci Series: 0 1 1 2 3

Aims:- Python program to perform string functions.

Procedure :-

String Functions

```
def add_string(str1):
    length = len(str1)
    if length > 1:
        if str1[-3:] == 'ing':
            str1 += 'ly'
        else:
            str1 += 'ing'
    return str1

print(add_string('do'))
print(add_string('according'))
```

Result: the program has been executed and output was verified.

- need to spend just at output cost with
output

doing

accordingly

($\frac{C}{C_0} = \frac{C_0 + \text{cost}}{C_0}$) \Rightarrow $C = C_0(1 + \frac{\text{cost}}{C_0})$

out

and

as much

function

$C = C_0(1 + \text{cost})$ \Rightarrow $C_0 = \frac{C}{1 + \text{cost}}$

$(C - \text{cost}) / C$ \Rightarrow $\frac{C - C_0}{C_0}$

$(C - C_0) / C_0 = \text{cost}$

\Rightarrow $\text{cost} = \frac{C - C_0}{C_0}$

cost

cost of

cost of

cost of

cost of

before last function and last mapping with function
before last

Program - 40

Aim: generate a list of four digit numbers in a given range with all their digits even and the number is perfect square.

Procedure :-

for i in range (1000, 10000, 1);

for j in range (30, 100, 1);

if $i == j^2$:

String = str(i)

if String \neq

if $\text{int}(\text{String}[0]) \% 2 == 0$ and $\text{int}(\text{String}[1]) \% 2 == 0$

and $\text{int}(\text{String}[2]) \% 2 == 0$ and $\text{int}(\text{String}[3]) \% 2 == 0$;

print(i)

Result: the program has been executed. The output was verified.

st component

maximum power output at component resist R_{load}

46.24	maximum power
60.84	
64.00	(ideal) power loss = 0
84.64	(ideal) power loss = 0

maximum power = $\frac{R_{load}}{R_{load} + R_s}$

power loss = $\frac{R_s}{R_{load} + R_s}$

(ideal) power loss = $\frac{R_s}{R_s + R_{load}}$

(ideal) power loss = $\frac{R_s}{R_s + R_{load}}$

(ideal) power loss = $\frac{R_s}{R_s + R_{load}}$

((ideal) power loss) limit

((ideal) power loss) limit

higher power requires more load component with $R_{load} \gg R_s$

Program :-

Aim :- Display the given pyramid with step number accepted from user.

Procedure :-

```
rows = int(input('enter the number of rows:'))  
for i in range(1,rows):  
    for j in range(i,i+1):  
        print(i*j,end=' ')  
    print()
```

Result: the program has been executed. the output was
verified.

Enter the number of rows: 6

1

2 4

3 6 9 (3) (3) (3) (3) (3) (3)

4 8 12 16 (4) (4) (4) (4) (4) (4)

(1) (1) (1) (1) (1) (1)

(1) (1) (1) (1) (1) (1)

10 digits have been written (5 digits) for p1

10 digits have been written (5 digits) for p2

(1) (1) (1) (1) (1)

The numbers with which each row is multiplied is 1 based

sequence

Aim :- Count the number of characters (Character Frequency) in a string.

Procedure :-

```
def char_frequency (str1):
```

```
    dict = {}
```

```
    for n in str1:
```

```
        keys = dict.keys()
```

```
        if n in keys:
```

```
            dict[n] += 1
```

```
        else
```

```
            dict[n] = 1
```

```
    return dict
```

```
print (char_frequency ("face book . com"))
```

and then how can we get output

$\{f:1, a:1, j:2, e:1, b:1, o:3, k:1, v:1, m:1\}$

- which of the following is correct
: (A) quick, (B) agrees with it
: (C) agrees with it very
(D) has, (E) having
(F) having

Aims:- Add 'ing' at the end of a given string. If it already end with 'ing' then add 'ly'

Procedure:

def add_string(str1):

length = len(str1)

if length > 2:

if str1[-3:] == 'ing':

str1 += 'ly'

else

str1 += 'ing'

return str1

print(add_string('crying'))

print(add_string('think'))

print(add_string('string'))

crying

thinking

stringy

Program :-

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

Procedure:-

```
def longestLength(a):
    maxi = len(a[0])
    temp = a[0]
    for i in a:
        if (len(i) > maxi):
            maxi = len(i)
            temp = i
```

print ("The word with the longest length is:",
 temp, "and length is", maxi)

a = input ("Enter a list elements separated by space")
a = a.split()
longestLength(a)

11) print elements separator will be 'for' like this:

Output:
Enter a list elements separated by space India is my country
the word with the longest is: country and length is 7

(longest) print like this

(longest) print - original

: < original list

: print - [8] like this

: print - 1 size

size

: print - 2 size

: print - 3 size

((longest)) print - like this list

((longest)) print - like this list

((longest)) print - like this list

Program: 25

Aim:- construct following pattern using nested loop.

Procedure:

```
Start : int (input ("Enter the number of rows :"))
For i in range (0,rows):
    For j in range (0,i+1):
        print ("*", end = " ")
    print (" ")
    For i in range (rows+1,0,-1):
        For j in range (0,i-1):
            print ("*", end = " ")
        print (" ")
End : }
```

Result:
The program has been executed and the output was
verified

Input output has been given as due to input signal
the output signal

output

(a) Multiplication

the factors of 232 are

1

2

4

8

29

58

116

$\underline{232}$

[a] is equal

and it is

(Greatest Common) 8

(Dad) = 1440

i = equal

at input is ~~1~~ with other both at $\underline{1}$ level

Output

(b) Multiplication

x

x x

x x x

x x x x x

x x x x

x x

x

Programs: 26

Aim:- generate all factors of a number

Procedure:-

```
def print_factors(x):  
    print ("The factors of", x, "are:")  
    for i in range(1, x+1):  
        if x % i == 0:  
            print (i)  
num = 10  
print_factors(num)
```

and value given earlier parallel numbers - with
output

the factors of 10 are:

- | | |
|----|--|
| 1 | ($1 \times 10, 10 \times 1$) equal to 10 |
| 2 | ($2 \times 5, 5 \times 2$) equal to 10 |
| 5 | ($5 \times 2, 2 \times 5$) equal to 10 |
| 10 | ($10 \times 1, 1 \times 10$) equal to 10 |

($1 \times 10, 10 \times 1$) speak of 1 unit

($2 \times 5, 5 \times 2$) speak of 1 unit

($5 \times 2, 2 \times 5$) 100%

($10 \times 1, 1 \times 10$) 100%

Programs: 27

Aim:- write lambda functions to find area of square, rectangle and triangle. ~~import~~

Procedure:

import math.

s_area = lambda len, ht : len * ht

t_area = lambda b, ht : b * ht / 2

c_area = lambda rad : math.pi * rad * rad

print ("area of rectangle (30,20) is", s_area(10,20))

print ("area of circle (15) is : ", c_area(15))

print ("area of triangle (12, 20) is", t_area(12, 20))

Result: the program has been executed and the output was verified.

radious per width no. decimal :- and
Output

area of rectangle (30, 20) is 200

area of circle (15) is : 1256.6370614359173

area of triangle (12, 20) is 220.0

: (15, 1) approx. at 1 sec

: 1256.6370614359173

(i) taking

01 = 0.001

(min) without taking

Program: 28

Aim: work with build-in packages.

Procedure:

```
import platform
```

```
x = platform.system()
```

```
print(x)
```

```
print()
```

```
x = dir(platform)
```

```
print(x)
```

```
print()
```

```
import datetime
```

```
x = datetime.datetime.now()
```

```
print(x)
```

```
x = datetime.datetime.now()
```

```
print(x.year)
```

```
print(x.strftime("%Y %A"))
```

Result: The program has been executed and the output was verified

sample of auto book of medical students done - with
highest suggested level of analysis

output

, report number	
2021-03-28	04:18:38 . 347291
2021-03-28	student = nato_3
2021-03-28	student = nato_3
sunday	student = nato_3
student = nato_3	student = nato_3
student = nato_3	student = nato_3
student = nato_3	student = nato_3

highly correlated with each other and may not be useful
- leading to overfitting

Program: Q9

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

Procedure:-

```
# Class Rect():
#     count = 0
#     def __init__(self, b, l):
#         self.b = b
#         self.l = l
#         self.l_count = self.l_count + 1
#     def area(self):
#         return self.b * self.l
#     def peri(self):
#         return 2 * (self.b + self.l)
#     if a.area == c.area:
#         print("equal")
#     else:
#         print("not equal")
# class Rectangle:
```

```
def __init__(self, length, breadth):
```

sey.length = length
sey.breadth = breadth

def area(sey):

return sey.length * sey.breadth

def peri(sey):

return 2 * (sey.breadth + sey.length)

a = int(input("Enter length of rectangle:"))

b = int(input("Enter breadth of rectangle:"))

c = int(input("Enter length of rectangle:"))

d = int(input("Enter length of rectangle:"))

obj = Rectangle(a,b)

obj1 = Rectangle(c,d)

print("Area of 1st rectangle:", obj.area())

print("Area of 2nd rectangle:", obj1.area())

print("Perimeter of 1st rectangle:", obj.peri())

print("Perimeter of 2nd rectangle is:", obj1.peri())

if obj.area() == obj1.area():

print("Equal")

else

print("not equal")

Result: the program has been ended. the output will be
verified.

equal rectangle over combined with a third rectangle
having base equal to width of rectangle having combined
area with output operation and consequent

Enter length of 1st rectangle : 5

Enter breadth of 1st rectangle : 10

Enter breadth of 2nd rectangle : 7

Enter breadth of 3rd rectangle : 10

Area of first rectangle : 50

Area of 2nd rectangle : 70

Perimeter of 1st rectangle : 30

Perimeter of 2nd rectangle : 34

not equal. (i.e. 30 & 34) so it is

Program: 80

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

Procedure:-

class Bank ():

def __init__(self):

self.balance = 0

print ("Your Account is Now Created")

def deposit(self):

amount = float (input("Enter the amount to be deposited:"))

self.balance += amount

print ("In your amount is deposited : ", amount)

def withdraw (self):

amount = float (input ("Enter the amount to be withdrawn:"))

if self.balance > amount:

 self.balance -= amount

 print ("In Your withdraw : ", amount)

else:

 print ("In insufficient balance")

def display (self):

 print ("In Net available Balance > ", self.balance)

s = Bank_Account()

s.deposit()

s.withdraw()

s.display()

Opener (Open) file
Account number - 1234

Output (File) name - abcd

Enter amount to be deposited : 900

Amount Deposited : 900.0

Enter amount to be withdrawn : 400

You withdrew : 400.0

Net available balance : 500.0

Operation to deposit : 1000.0

Operation to withdraw : 500.0

(File) deposited = 1000

(File) withdrawn = 500

(File) available = 500

Amount deposited : 1000.0

Amount withdrawn : 500.0

Amount available : 500.0

(File) deposit

(File) withdraw

(File) available

Aim: Create a class Rectangle with private attribute length and width and add 'c' to compare the area of two rectangles
 Procedure:

Class A :

- length = 0
 - width = 0
 - area = 0

def __init__(self, l, w):

self.length = l

self.width = w

def area(self):

self.area = self.length * self.width

def gt(self, other):

if (self.area > other.area):

return True

else:

return False

test1 = A(3,4)

test1.area()

test2 = A(6,5)

test2.area()

if (test1 > test2):

print ("test1 is greater than test2")

else:

print ("test2 is greater than test1")

output

rect 2 is greater than rect 1

(prob) - just 1 prob

or = exactly prob

but not at least one of them is true

(prob) depends on

which known value of θ is true or false

(θ) depends

known or not exactly prob

known or not known, not all of them are

(prob) exactly true prob

known or value of θ is true known

(θ) exactly true prob

known or exactly true prob

known or not exactly true prob

known or not known not all of them

(prob) exactly true prob

(prob) depends on

whether you're exactly right or not all of them

(θ) depends on

Program: 8.2

Aim: Create a class Time with private attributes hour, minute and second. overload '+' operator to find sum of two time and print result.

Procedure: Define add() and calculate sum

class Time:

def __init__(self, hour, minute, second):

self.hour = hour

self.minute = minute

self.second = second

def add(self, other):

h = self.hour + other.hour

s = self.second + other.second

self.calculate(h, m, s)

def calculate(self, h, m, s):

s = s

h = h

m = m

if s > 60:

m = m + 1

if s > 60:

M = M + 1

s = s - 60

if m > 60:

h = h + 1

if m > 60:

h = h + 1

$$NO = m - 60$$

Print ('NEW TIME')

Print ('{0} : {1} : {2}'. format(h, m, s))

Return 0

time1 = input ('Enter first time in the format HH:MM:SS\n')

time2 = input ('Enter second time in the format HH:MM:SS\n')

h1, m1, s1 = map (int, time1.split (:))

h2, m2, s2 = map (int, time2.split (:))

t1 = Time (h1, m1, s1)

t2 = Time (h2, m2, s2)

t1 + t2

Result: the program has been exectl. The output was
verified.

Output: will be displayed in

Enter first time in the format HH:MM:SS

0:0:3

Enter second time in the format HH:MM:SS

4:58:59

(Current, end, start) - 1st

NOW TIME

Current - 2nd - 1st

7:1:2 (Current, Start - 1st)

7:1:2 (Current, End - 1st)

7:1:2 (Current, Start - 2nd)

7:1:2 (Current, End - 2nd)

7:1:2 (Current, Start - 3rd)

7:1:2 (Current, End - 3rd)

7:1:2 (Current, Start - 4th)

7:1:2 (Current, End - 4th)

7:1:2 (Current, Start - 5th)

7:1:2 (Current, End - 5th)

7:1:2 (Current, Start - 6th)

7:1:2 (Current, End - 6th)

7:1:2 (Current, Start - 7th)

7:1:2 (Current, End - 7th)

7:1:2 (Current, Start - 8th)

7:1:2 (Current, End - 8th)

Program : 33

Aim :- Create a class Publisher Derive class book from publisher with attributes from book attributes price and no. of pages. write a program that display information about a python book use base/child inheritance and multiple inheritance.

Procedure :-

Class Publisher:

def __init__(self, pubname):

self.pubname = pubname

def display(self):

print("Publisher Name:", self.pubname)

class Book(Publisher):

def __init__(self, pubname, title, author):

Publisher.__init__(self, pubname)

self.title = title

self.author = author

def display(self):

print("Title:", self.title)

print("Author:", self.author)

class Python(Book):

def __init__(self, pubname, title, author, price, no_of_pages):

Book.__init__(self, pubname, title, author)

self.price = price

self.no_of_pages = no_of_pages

def display(self):

print("Title:", self.title)

print("Author:", self.author)

```
print ("price:", self. price)
print ("number of pages", self. no_of_page)
s1 = python ("ak books", "Taming Python By Programming",
             'Java Jose', 200, 219)
s1. display()
```

Result: the program has been exectd. the output was
verified.

Output

Title : Taming Digits by Programming

Author : John Jose, Date : 1/1/08

Pno. : 866

Number of Digits : 219

(Digit by digit, starting from left)

Digit by digit, starting from right

Programs : 34

Aim: Write a python program to read a file line by line and store it into a list

Procedure :

[Illustration]

```
str1 = "welcome to python programming\n" + "python" + "\n"
fw = open ("Afile.txt", "w")
fw.write (str1)
fw.close ()

fr = open ("Afile.txt", "r")
str2 = fr.readlines ()
for i in str2:
    print (i)
```

Result: The program was executed and has been successfully verified.

(output for `print("Hello")`) would
be "Hello".
Output
welcome to python programming
Python

Aim: Python programs to copy odd lines of one file to another.

Procedure :

```

fo = open ('file1.txt', 'r')
str1 = fo.readline()
fo.close()

fo = open ('file2.txt', 'w')
x = 0
for i in str1:
    x = x + 1
    if x % 2 != 0:
        fo.write(i)
fo = open ('file2.txt', 'r')
str2 = fo.readline()
print(str2)

```

Result: the program was successfully executed and the result has been verified.

Output

[char(111)]

Good done

Aim: Write a python program to read each row in a given csv file and print or list as strings.

Procedure:

```
import csv
with open ('movie1.csv', 'w', newline = '') as file:
    writer = csv.writer(file)
    writer.writerow(['SN', 'Movie', 'Rating'])
    writer.writerow([1, 'Lord of Rings', 5])
    writer.writerow([2, 'Harry Potter', 6])
with open ('movie1.csv') as csv_file:
    data = csv.reader(csv_file)
    for row in data:
        print(','.join(row))
```

Result: the above program was executed and the result was obtained.

Output

SA , movie , Rating

1. Lord of the Rings . 5
2. Harry Potter

Program: 37

Aim: Write a python program to read specific columns of a given csv file and print the content of the columns.

Procedure:

```
import csv  
with open ('fruits1.csv','w', newline= '') as file :  
    write = csv.writer(file)  
    write.writerow(['sno', 'Fruit', 'Rate'])  
    write.writerow(['1', "apple", "60"])  
    write.writerow(['2', "orange", "55"])  
    write.writerow(['3', "grapes", "70"])  
    write.writerow(['4', "banana", "85"])  
  
with open ('fruits1.csv','r') as file:  
    data = csv.reader(file)  
    print("Content in column fruits:")  
    for x in data:  
        print(x[1])
```

Result : the above programs was successfully executed and the result received.

Output

Content in the column 'fruit':

Fruit

apple

orange

grape

banana

Program: 38

Aims: Write a Python program to write a dictionary to a CSV file. After writing the CSV file read the CSV file and display the content.

Procedure:

import csv

f = open("fruits.csv", "w")

writer = csv.DictWriter(f, fieldnames=["fruit", "count"])

writer.writeheader()

writer.writerow(2 "fruit": "Apple", "count": "1")

writer.writerow(2 "fruit": "Banana", "count": "2")

f.close()

c = 0

f = open("fruits.csv")

reader = csv.DictReader(f)

for row in reader:

if c == 0

print f[" ".join(row)]

print f["row("fruit")"], f["row("count")"]

f.close()

Result:- The program was successfully executed and the result was worked.

Output

Front	count
Apple	1
Banana	2

(apple, orange, banana, watermelon, lemon)
apple (apple, orange, banana, watermelon, lemon)
(apple, orange, banana, watermelon, lemon)
(apple, orange, banana, watermelon, lemon)
(apple, orange, banana, watermelon, lemon)

Program: 39

Aim: Create a package graphics with module rectangle, circle and such package 3d - graphics with module (cuboid and sphere) include method to find area and perimeter of different respective figures in each module. write program that finds area and perimeter of figures by different importing statements.

Procedure:

```
from graphics import *
```

```
print("Rectangle")
```

```
l = int(input("Enter length :"))
```

```
b = int(input("Enter breadth :"))
```

```
print("Area of Rectangle : " + str(area(l,b)))
```

```
print("Perimeter of Rectangle : " + str(perimeter(l,b)))
```

```
print("Circle")
```

```
r = int(input("Enter a radius :"))
```

```
print("Area of Circle : " + str(area(r)))
```

```
print("Circumference of Circle : " + str(perimeter(r)))
```

```
print("Sphere")
```

```
r = int(input("Enter a radius :"))
```

```
print("Area of Sphere : " + str(area(r)))
```

```
print("Surface area of Sphere : " + str(surface_area(r)))
```

```
print("Cuboid")
```

```
l = int(input("Enter a length :"))
```

```
w = int(input("enter a width"))
```

```
h = int(input("enter a height"))
```

```
print("area of cuboid : ", area(l,w,h))
```

```
print("perimeter of cuboid : ", perimeter(l,w,h))
```

Circle

```
from math import pi
```

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

```
    return (pi * r * r)
```

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

```
    return (2 * pi * r)
```

Rectangle

```
def area(l, b):
```

```
    return (l * b)
```

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

```
    return (2 * l + b))
```

Sphere

```
from math import pi
```

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

```
    return (4 * pi * r * r)
```

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

```
    return ((4/3) * pi * r * r * r))
```

Cuboid

```
def area(l, w, h):
```

```
    return ((2 * l * w) + (2 * l * h) + (2 * w * h))
```

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

```
    return (4 * (l + w + h))
```

Result: The program was executed and the output has been verified.

Output

Rectangle

Enter Length : 5

Enter Breadth : 10

Area of rectangle : 50

Perimeter of rectangle : 30

Circle

Enter Radius of Circle : 4

Area of Circle : 50.26548245

Perimeter of Circle : 25.132741728

Sphere

Enter the radius of sphere : 3

Area of Sphere : 113.0973352923255

Perimeter of sphere : 113.0973355292325

Cuboid

Enter Length : 7

Enter width : 12

Enter height : 8

Area of cuboid : 472

Perimeter of cuboid : 108.