

Date: 13/01/2021

Program No: 1

AIM: Write a python program to find the biggest of three numbers entered by the user.

Program:

```
x = int(input("Enter the first Number:"))
y = int(input("Enter the second Number:"))
z = int(input("Enter the third Number:"))

if (x>y) and (x>z)
    largest = x
elif (y>x) and (y>z)
    largest = y
else:
    largest = z

print("The largest Number is", largest)
```

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

Output

Enter the First Number : 5

Enter the Second Number : 8

Enter the Third Number : 4

The largest Number is : 8

Date: 13/01/2021

Program NO: 2

AIM: Write a python program to find a square of a number entered by the user.

Program:

```
a = int(input("Enter the Number :::"))
sqr = a * a
print("the square of the number is :::{0}
      ={1}".format(a,sqr))
```

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

Output

Enter -the Number :::: 5
the square of -the number is :: 5=25

Date : 13/01/2021

Program Number : 3

AIM: Write a python program to return the area of a circle using a Function.

Program :

```
import math
```

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

```
area=math.pi*r*r
```

```
print("%2f"%area)
```

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

Output

Area covered

Enter the radius of the circle : 5

78.54

The area covered by the circle is 78.54 square units.

Area = πr^2 .

Radius of the circle is 5 units. Area = πr^2 .
Area = $\pi \times 5^2$.
Area = 25π .
Area = 25×3.14 .
Area = 78.54.

Radius need to change soil. If we increase
radius even more soil will be covered.

Date : 25/10/2021

Program NO: 4

AIM: Form a list of vowels selected from a given word.

Program :

```
def checkvow(string, vowels)
```

```
    final = [each for each in string if  
             each in vowels]
```

```
    print(len(final))
```

```
    print(final)
```

String = "I am a Student"

Vowels = "AaEeIiOoUu"

```
checkvow(string, vowels)
```

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

Output

5

[‘I’, ‘a’, ‘a’, ‘u’, ‘e’]

Date: 25/1/2021

Program NO:5

AIM: Count -the occurrences of each word
in a line of text.

Program:

```
def wordcount(str):  
    counts = dict()  
    words = str.split()  
    for word in words:  
        if word in counts:  
            counts[word] += 1  
        else:  
            counts[word] = 1  
    return counts
```

```
Print(wordcount('I am a student. I am a  
programmer'))
```

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

Output

student : 1 :: 1,
xi : 2, am : 2, a : 1, "programmer" : 13

(elsewhere) was selected

an example will show the mechanism

Example of class

Parent Class

Child Class

"derivative of no 1" = Bank

"sub-class" - class w/

columns, first row (elsewhere)

first class contains only 1 value

last row: empty cell - bank

Date: 25/1/2021

Program No: 6

AIM: Store a list of first names. Count the occurrences of 'a' within the list.

Program:

```
def countx(lst, x):  
    count=0  
    for ele in lst:  
        if (ele==x):  
            count=count+1  
    return count
```

lst = ['a', 'v', 'c', 'a']

x='a'

Print ('{} has occurred {} times'.format
(x, countx(lst, x)))

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

Output

a has occurred 2 times.

Date: 25/1/2021

Program NO: 7

AIM: Enter two lists of integer. check whether list are of same length, check whether lists sums to same value, check whether any value occur in both.

Program:

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

```
list2 = [1, 2, 3, 4, 5]
```

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

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

```
if (len1 == len2):
```

```
    print('both the list are equal length')
```

```
else:
```

```
    print('both the list are not equal  
length')
```

```
tot1 = sum(list1)
```

```
tot2 = sum(list2)
```

```
if (tot1 == tot2):
```

```
    print("sum of the both the list equal")
```

```
else:
```

```
    print("sum of the both the list not equal")
```

for value in list1:

 if value in list1:

 common = 1

 if common == 0:

 print("there are common element")

else:

 print("no common element")

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

Output

Output

Both the list is equal length

Sums of the both list not equal

No common element.

None

Program NO: 8

Date: 25/01/2021

AIM: Get a string from an input string where all occurrences of first character replaced with '\$', except first character
[Eg: onion → oni\$n]

Program:

```
def charchange(str1):
```

```
    char = str1[0]
```

```
    str1 = str1.replace(char, '$')
```

```
    str1 = char + str1[1:]
```

```
    return str1
```

```
Print(charchange('onion'))
```

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

Output

Onishu

Output of subproblem

of connection 21

Book of this one is

1962

Line ("of common sense

Replies back and forward
Output mode available

Date: 25/1/2021

Program NO:9

AIM: Create a string from given string where first and last character changed

[Eg: Python → ngthon]

Program:

```
'def changestring(stri):  
    return stri[-1] + stri[1:-1] + stri[0]  
print(changestring('Python'))'
```

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

output

BESTE WISSENSCHAFT FÜR DEIN LEBEN

[RECORD NO. 11183]

Chippewa

(1342) ~~spinosus~~ 735

$$\text{Earnings} = 15 \times 15$$

Ques 1. $\text{मात्रा } \text{अवधि} = 17.2$

$$E[18d^2 + 180d^3] = 18/8$$

1848 (1853)

Group of Subspecies (Species)

Date : 25/1/2021

Program NO: 10

AIM: Accept an integer n and compute

$$\bullet n + nn + nnn.$$

a = int(input("Input an integer : "))

$n_1 = \text{int}(\% s \% a)$

$n_2 = \text{int}(\% s \% s \% (a, a))$

$n_3 = \text{int}(\% s \% s \% s \% (a, a, a))$

Print ($n_1 + n_2 + n_3$)

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

Output

Input an integer: 6

738

Output error message
Address ends digit is not

Program NO:11

Date: 25/1/2021

Aim: Sort dictionary in ascending
and descending order.

Program:

```
y = {'orange': 3, 'apple': 2, 'mango': 13}
```

```
l = list(y.items())
```

```
l.sort()
```

```
print('Ascending order is ', l)
```

```
l = list(y.items())
```

```
l.sort(reverse=True)
```

```
print('Descending order is ', l)
```

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

Output

Ascending order is [('apple', 2), ('mango', 1), ('orange', 3)]

Descending order is [('orange', 3), ('mango', 1), ('apple', 2)]

Date: 25/11/2021

Program NO: 12

AIM: Merge -two dictionaries.

Program:

```
fruits = {"apple": "red", "orange": "orange", "tangerine": "orange"}  
dryfruits = {"cashew": "brown", "almond": "brown", "pistachio": "green"}
```

```
fruits.update(dryfruits)
```

```
print(fruits)
```

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

Output

```
{'cashew', 'almond', 'apple', 'orange',  
'tangerine', 'pistachio'}
```

Date: 25/1/2021

Program No: 13

AIM: Find gcd of two numbers.

Program:

```
def gcd(a,b):  
    if (b == 0):  
        return a  
    else:  
        return gcd(b, a%b)
```

a=60

b=48

Print ("the gcd of 60 and 48 is : ", end = " ")

Print (gcd(60,48))

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

Output

The gcd of 60 and 48 is: 12

Date: 25/1/2021

Program NO: 14

AIM: From a list of integers, create a list removing even numbers.

Program

```
list = [11, 22, 33, 44, 55]
```

```
Print(list)
```

```
for i in list:
```

```
    if (i % 2 == 0):
```

```
        list.remove(i)
```

```
Print("list after removing even numbers:",  
      list)
```

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

Output

[11, 22, 33, 44, 55]

list after removing even numbers:

[11, 33, 55]

Date: 27/1/2021

Program No:15

Aim: Program to find -the factorial of a number.

Program:

```
def factorial(n):
```

```
    if(n==0)
```

```
        return 1
```

```
    return n*factorial(n-1)
```

```
num=5
```

```
Print ("Factorial of ", num, "is", factorial(num))
```

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

Output

Factorial of 5 is 120

(5!) = 120

Product of 5 is 120

120 = 120

120 = 120

Product of 5 is 120
Product of 5 is 120

(5!)

Product of 5 is 120
Product of 5 is 120

Program No: 15

Date: 27/1/2021

AIM: Generate Fibonacci Series of N terms.

Program:

```
def fibo(n):
```

```
    if n <= 1:
```

```
        return n
```

```
    else:
```

```
        return (fibo(n-1) + fibo(n-2))
```

```
nTerms = int(input("Enter the value of n:"))
```

```
if nTerms <= 0:
```

```
    print("No Fibonacci Sequence :")
```

```
else:
```

```
    print("Fibonacci Sequence :")
```

```
    for i in range(nTerms):
```

```
        print(fibo(i))
```

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

Output

To demonstrate - output of a user-defined function
Enter the value of 'n': 5

Fibonacci sequence:

0
1
1
2
3

Program NO: 17

Date: 27/1/2021

AIM: Find the sum of all items in a list.

Program:

def sumlist(items):

 sumnumbers = 0

 for x in items:

 sumnumbers = sumnumbers + x

 return sumnumbers

Print(sumlist([1, 2, -12]))

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

output

-9

negative number

negative

number is negative

10^{-10}

negative

number

negative number is negative

negative number is negative

number is negative

number is negative

number is negative

number is negative

number is negative

number is negative

number is negative

negative number is negative

Program NO: 18

Date: 27/1/2021

AIM: Generate a list of four digit number
in a given range with all their
digits even and the number is a
perfect square

for p in range(1000, 10000):

 for j in range(32, 99):

 if p == j*j:

 num = str(p)

 if (int(num[0])%2==0 and int(num[1])%2==0 and int(num[2])%2==0 and int(num[3])%2==0):

 print(p)

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

Output

4624

6084

6400

8464

28 + 65 = 93 \rightarrow eigentl. 93

Ergebnis nicht pass.

((LS-EU(7))-durch) \rightarrow 1

bed. max. dass es sich umgängt soll: 131,155
bed. max. dass es sich umgängt soll: 131,155

Date: 27/1/2021

Program No: 1819

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

Eg: N=4

1
2 4
3 6 9
4 8 12 16

Program:

rows=5

for i in range(1, rows):

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

 print(i*j, end='')

 print()

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

Output

திட்டங்கள் என்று

1	2	4	8	16	32	64	128	256	512	1024	2048	4096
1	2	4	8	16	32	64	128	256	512	1024	2048	4096
1	2	4	8	16	32	64	128	256	512	1024	2048	4096
1	2	4	8	16	32	64	128	256	512	1024	2048	4096
1	2	4	8	16	32	64	128	256	512	1024	2048	4096

(0000), 0010 போன்ற நிலை

(FF, FF) போன்ற நிலை

$$2^{12} = 4096$$

$$2^{12} = 4096$$

போன்ற நிலை (0000) 0000

போன்ற நிலை 0000 (FF, FF)

போன்ற நிலை 0000 0000

(1) கணக்கு

போன்ற நிலை போன்ற நிலை 0000 0000

போன்ற நிலை போன்ற நிலை 0000 0000

Date: 27/1/2021

Program NO: 1920

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

Program:

```
def charfrequency(str1):
    dict = {}
    for n in str1:
        keys = dict.keys()
        if n in keys:
            dict[n] = dict[n] + 1
        else:
            dict[n] = 1
    return dict.
```

```
print(charfrequency('frequency of character'))
```

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

Output

```
{'f': 2, 'x': 3, 'e': 3, 'q': 1, 'u': 1, 'n': 1,  
'c': 3, 'g': 1, ' ': 2, 'o': 1, 'h': 1, 'a': 2,  
't': 13}
```

Program No: 2021

Date: 27/1/2021

Aim: Add 'ing' at the end of a given string. If it already ends with 'ing', then add 'ly'.

Program:

```
def addstr(str1):
    length = len(str1)
    if length > 2:
        if str1[-3:] == "ing":
            str1 += "ly"
        else:
            str1 += "ing"
    return str1
```

```
Print(addstr ("good"))
Print (addstr ("morning"))
```

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

Output

googling

Movingly

Output = Input

Input = Output

Input = Output

Output

Input

Output

(Outputs to process (Input) + all)

Input and output self linking
but not even higher self link

Program No: 22

Date: 27/1/2021

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

Program:

```
def longestword(wlist):
    word_len = []
    for n in wlist:
        word_len.append([len(n), n])
    word_len.sort()
    return word_len[-1][0], word_len[-1][1]
```

```
result = longestword(["apple", "mango", "Banana"])
```

```
print("In longest word:", result[1])
```

```
print("length of the longest word:", result[0])
```

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

Output

Longest word: banana

Length of the longest word: 6

Output

:("apple") apple 71

:("dog") dog 41

:("cat") cat 31

:("banana") banana 71

:("pig") pig 41

:("dog") dog 41

1234 5678

:("boob") boob 51

:("banana") banana 71

Max's code will output all the words in the list as single self-terms

Max's code will output all the words in the list as single self-terms

Program NO: 22

Date: 27/1/2021

AIM: Construct Following pattern using
nested loop: ~~isolate + self - is K+B~~

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

Program:

$n=5$

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

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

Output

* *

六

文獻卷

水 水 水 水

中華書局影印

4. \rightarrow 5.

中華書局影印

四

卷之三

卷之三

100123

100000

卷之三

6

卷之三

1-3 31108

20-0603

605

卷之三

Date: 27/11/2021

Program NO: 24.

Aim: Generate all factors of a number.

Program:

def factors(x):

 Print("The factors of ", x, "are:")

 For i in range(1, x+1):

 If x % i == 0:

 Print(i)

num = int(input("Enter the number:"))

factors(num)

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

Output

Enter the number : 10

Factor of 10 are :

1
2
5
10

Date: 27/11/2021

Program NO: 25

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

Program:

Square = lambda a: a*a

Print(Square(5))

rectangle = lambda w,h : w*h

Print(rectangle(5,3))

triangle = lambda b,h : .5*b*h

Print(triangle(5,3))

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

Output

25

15

7.5

(192 x 192) image of 3D

: 0.0 - 9.5 - 21

0.0 - 9.5 - 21

(192 x 192) image of 3D

Date: 27/2/21

Program NO: 26.

AIM: Python program to create a package graphics with modules rectangle, circle and sub package 3D graphics with modules cuboid and sphere. Include methods to find area and perimeter of respective figures in each module.

Write programs that find area and perimeter of figures by different importing statements.

Program:

Circle.py:

def area(r):

print('Area of circle', r, '%.2f' % (3.14 * r * r))

def circumference(r):

print('circumference of circle', r, '%.2f' % (3.14 * 2 * r))

rect.py

def area(a,b):

 print('Area of rectangle', a, 'and', b, '% of
 %', (a*b))

def perimeter(a,b):

 print('Perimeter of rectangle', a, 'and',
 b, '% of' % (2*(a+b)))

Sphere.py

def area(r):

 print('Area of sphere', r, '% of' % (4 *
 3.14 * r * r)))

def perimeter(r):

 print('Perimeter of sphere', r, 'is', '%
 of' % (2 * 3.14 * r))

Cuboid.py

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

```
    print('area of cuboid',l,',',b,',',h,',',is:',  
          '%.2f' %(2*((l+b)+(b+h)+(l+h))))
```

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

```
    print('Perimeter of cuboid',l,',',b,  
          ',',h,',', '%.2f' %(4*(l+b+h)))
```

Perimeter.py

```
import circle
```

```
from rect import *
```

```
from graphics...3Dgraphics import cuboid,sphere
```

```
l1 = float(input('Enter length of the rectangle :'))
```

```
b1 = float(input('Enter breadth of the rectangle :'))
```

```
perimeter(l1,b1)
```

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

```
ci = circumference(r)
```

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

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

```
b2 = float(input('Enter height of the cuboid:'))  
cuboid.area(l2,b2,h2)  
r = float(input('Enter the radius of the sphere:'))  
sphere.perimeter(r)
```

Area.py

```
import circle
```

```
from rect import *
```

```
from graphics import cuboid,  
sphere
```

```
l1 = float(input('Enter the length of the rectangle:'))
```

```
b1 = float(input('Enter the breadth of the rectangle:'))
```

```
area(l1,b1)
```

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

```
circle.area(r)
```

```
l2 = float(input('Enter the length of the cuboid:'))
```

```
b2 = float(input('Enter the breadth of the cuboid:'))
```

```
h2 = float(input('Enter the height of the cuboid:'))
```

```
cuboid.area(l2,b2,h2)
```

```
r=float(input('Enter the radius of the sphere:'))  
sphere.area(r)
```

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

```
100.0  
12566.37  
12566.37  
12566.37  
12566.37
```

Output

Enter length of the rectangle: 4

Enter breadth of the rectangle: 3

Perimeter of rectangle 4.0 and 3.0 is
14.00

Enter the radius of the circle: 2

Circumference of circle 2.0 is 12.56

Enter length of the cuboid: 5

Enter breadth of the cuboid: 4

Enter height of the cuboid: 3

Perimeter of cuboid 5.0, 4.0, 3.0 is
48.00

Enter the radius of the sphere: 2

Perimeter of sphere is 2.0 is 12.56

Enter length of -the rectangle : 2

Enter breadth of -the rectangle : 3

Area of rectangle 2.0 and 3.0 is 6.0

Enter the radius of -the circle : 4

Area of circle 4.0 is 50.24

Enter length of the cuboid : 4

Enter breadth of the cuboid : 7

Enter height of the cuboid : 2

Area of cuboid 4.0, 7.0, 2.0 is 100.00

Enter the radius of the sphere : 1

Area of sphere 1.0 is 12.56

Program NO: 27

Date: 27/2/2021

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

Program:

import datetime

a = datetime.datetime.now()

a = a.year

b = int(input("Enter the Final year"))

print("leap years")

for i in range(a, b+1):

if (i % 4 == 0)

print(i)

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

Output

Enter the final year: 2040

leap years: 2024, 2028, 2032, 2036, 2040.

2024

2028

2032

2036

2040.

Program No: 28

Date: 27/2/21

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

Program:

list = [10, -20, 0, 30, -40, 50]

for num in list:

 if num >= 0:

 print(num)

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

Output

most cases good enough (approx 10-15% error) and very efficient

0

30

50

70

90

110

130

150

170

190

210

230

250

270

290

310

330

350

370

390

410

430

450

470

490

510

530

550

570

590

610

630

650

670

690

710

730

750

770

790

810

830

850

870

890

910

930

950

970

990

1010

1030

1050

1070

1090

1110

1130

1150

1170

1190

1210

1230

1250

1270

1290

1310

1330

1350

1370

1390

1410

1430

1450

1470

1490

1510

1530

1550

1570

1590

1610

1630

1650

1670

1690

1710

1730

1750

1770

1790

1810

1830

1850

1870

1890

1910

1930

1950

1970

1990

2010

2030

2050

2070

2090

2110

2130

2150

2170

2190

2210

2230

2250

2270

2290

2310

2330

2350

2370

2390

2410

2430

2450

2470

2490

2510

2530

2550

2570

2590

2610

2630

2650

2670

2690

2710

2730

2750

2770

2790

2810

2830

2850

2870

2890

2910

2930

2950

2970

2990

3010

3030

3050

3070

3090

3110

3130

3150

3170

3190

3210

3230

3250

3270

3290

3310

3330

3350

3370

3390

3410

3430

3450

3470

3490

3510

3530

3550

3570

3590

3610

3630

3650

3670

3690

3710

3730

3750

3770

3790

3810

3830

3850

3870

3890

3910

3930

3950

3970

3990

4010

4030

4050

4070

4090

4110

4130

4150

4170

4190

4210

4230

4250

4270

4290

4310

4330

4350

4370

4390

4410

4430

4450

4470

4490

4510

4530

4550

4570

4590

4610

4630

4650

4670

4690

4710

4730

4750

4770

4790

4810

4830

4850

4870

4890

4910

4930

4950

4970

4990

5010

5030

5050

5070

5090

5110

5130

5150

5170

5190

5210

5230

5250

5270

5290

5310

5330

5350

5370

5390

5410

5430

5450

5470

5490

5510

5530

5550

5570

5590

5610

5630

5650

5670

5690

5710

5730

Program NO: 28

Date: 27/2/2021

Aim: Accept a file name from user
and print extension of that.

Program:

```
filename = input("Input the filename:")
```

```
F_extns = filename.split(".")
```

```
print("The extension of the file is: " +
```

```
sep(F_extns[-1]))
```

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

Output

Input - the file name: file.txt

The extension of the file is: 'txt'

$$[0.01-0.08, 0.05-0.1] = 0.01$$

It is a decimal number not

a fraction (order 3)

(minus) sign

minus sign and compare self (these)
but now, suppose self (bars)

Date: 27/2/2021

Program NO: 30

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

Program:

```
values = input("Input some comma  
separated color :")  
list = values.split(",")  
print("%s %s" % (list[0], list[-1]))
```

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

Output

Input some comma separated color:

red, black, Orange.

red, orange.

(C:\Users\sonash\Downloads\color)

red black orange

((color))

Pass and output will be like this

red black orange

Date: 14/2/2021

Program NO: 3

Aim: Print out all colors from color-list1
not contained in color list 2.

Program :

```
color-list-1 = set(["white", "Black", "Red",  
"gray"])
```

```
color-list-2 = set(["Red", "Green", "orange"])
```

```
Print(color-list-1.difference(color-list-2))
```

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

Output

Output to file in terms of
{'Black', 'gray', 'white'}

def output_file(file_name):

for row in image:

for col in row:

if col == 0:

file_name.write("black")

elif col > 0 and col < 128:

file_name.write("gray")

else:

file_name.write("white")

file_name.close()

def main():

image = [[0 for i in range(128)] for j in range(128)]

Date: 27/2/2021

Program No: 32

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

Program:

```
a = input("Enter the first string:")
b = input("Enter the second string:")
x = a[0:1]
a = a.replace(a[0:1], b[0:1])
b = b.replace(b[0:1], x)
print(a, b)
```

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

output

enter the first string: good

enter the second string: Morning

mood gaining:

good mood - bad mood
(I gain)

bad mood - good mood

good mood - bad mood

good mood - bad mood

how does it express self-harm

Program NO: 32

Date: 17/2/2021

AIM: Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare two rectangle object 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 2 * (self.length + self.breadth)

def comparison(self, obj):

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

Print('The highest

area of rectangle with

length = ', self.length, ' and breadth =
, self.breadth)

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

Print('The highest area
of rectangle with length = ', obj.
length, ' and breadth = ', obj.
breadth)

else:

Print('They have equal area')

$x_1 = \text{Rectangle}(10, 20)$

$x_2 = \text{Rectangle}(50, 60)$

$x_1 < x_2$

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

output

The highest area of Rectangle with length = 50 and breadth $b=60$.

— 3 —

[1:0] 53 - 2

(d, d) Acne

Date: 27/2/21

Program NO: 34

AIM: Create a bank account with member 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 Account:

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

self.no = a

self.name = n

self.type = t

self.balance = b

def deposit(self, a):

self.balance = self.balance + a

print('Rs.', a, 'deposited,
current balance is: Rs.',

```
self.a = - )  
def withdraw(self, a):  
    if self.balance >= a:  
        self.balance = a  
        print('Rs.', a, 'withdraw,  
        current balance is: Rs.', self.  
        a)  
    else:  
        print('Insufficient balance  
        to make this transaction')  
a = int(input('Enter account number:'))  
n = int input('Enter the name of the  
account holder: ')  
t = input('Enter account type')  
b = float(input('Enter your balance'))  
aci = Account(a, n, t, b)  
aci.deposit(float(input('Enter amount  
to deposit :')))  
aci.withdraw(float(input('Enter amount  
to withdraw :')))
```

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

Output

Enter account number: 1200637
Enter name of the account holder: Aa
Enter account type: current
Enter your balance: 20000
Enter amount to deposit: 1500
Rs. 1500.00 deposited, current balance is:
Rs: 21500
Enter amount to withdraw: 1500
Rs. 1500 is withdrawn, current balance:
Rs. 20000

Program NO: 35

Date: 27/2/2021

AIM: Create a class Rectangle with private attributes length and width. overload ' $<$ ' operator to compare the area of 2 rectangles.

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('The lesser area rectangle with length = ', self.length, 'and width = ', self.width)

elif other.area < self.area:

Print('The lesser area Rectangle with length = ',

other.length, 'and width = other.
width)

else:

Print('They have equal area')

Rect 1 = Rectangle(l, w)

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

w = float(input('Enter width of the
second rectangle'))

Rect 2 = Rectangle(l, w)

Rect1 < Rect2

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

Output

Enter length of the first rectangle: 30

Enter width of the first rectangle: 20

Enter length of the second rectangle: 50

Enter width of the second rectangle: 10

The lesser area rectangle with length
= 30 and width = 20

Date: 27/2/21

Program NO: 36

AIM: Create a class Time with private attributes hour, minute and second. Overload '+' operator to find sum of 2 times.

Program:

class Time:

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

self.hour = h

self.minute = m

self.second = s

def __add__(self, other):

hour = int((self.hour + other.hour)

% 24 + (self.minute + other.minute)

/ 60) minute = int((self.minute

+ other.minute) % 60 + ((self.

second + other.second) / 60))

second = int((self.second +

other.second) % 60)

Print ('Time = ', hour, ' hour:', minute,
' minut:', second, 'second')

T₁ = Time(10,20,15)

T₂ = Time(16,45,56)

T₁ + T₂

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

output

Time = 3 hours 6 minute 11 second

Jump shot (soft) 30cm
Jump shot (soft) 35cm 1.5 sec
Jump shot (soft) 40cm 1.5 sec
Jump shot (soft) 45cm 1.5 sec
Spike shot (soft) 30cm 1.5 sec
Spike shot (soft) 35cm 1.5 sec
Spike shot (soft) 40cm 1.5 sec
Spike shot (soft) 45cm 1.5 sec

Best and majority soft ball hand
Volleying soft ball

Date: 27/2/2021

Program NO: 34

AIM: Create a class publisher(name).

Derive class book from publisher
with attributes title and author.

Derive class Python from book
with attributes price and no-of-
Pages. Write a program that
displays information about a
Python book. Use base class
constructor invocation and method
overriding.

Program:

class publisher:

 def __init__(self, name):

 self.name = name

 def show(self):

 pass

class Book(Publisher):

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

self.title := title

self.author = author

Publisher.__init__(self, name)

def show(self):

pass.

class Python(Book):

def __init__(self, p, no, title, author, name):

self.price = p

self.no_of_pages = no

Book.__init__(self, title,

author, name)

def show(self):

print('Title of the book:',
self.title)

print('Author of the book:',
self.author)

print('Publisher of the
book:', self.name)

```
print('price of the book : 'self.price)  
print('no. of pages in the book : '  
      self.no_of_page)  
  
p1 = Python(400, 100, 'python', 'G V Rossen',  
            'books pvt')  
  
p1.show()
```

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

output

Title of the book: Python

Author of the book: G V Rossum

Publisher of the book: book pvt.

Price of the book = 400

No. of pages in the book: 1000

thus each page will be

Date : 21/2/2021

Program NO: 38

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

Program:

```
def file_read(fname):  
    with open(fname) as f:  
        c = f.readlines()  
        print(c)  
        print(len(c))  
file_read ("demo.txt")
```

Text file: demo.txt

This is a python program
to read line by line
and store it into a list

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

Output

['This is a python program to ' to
read a file line by line ' and
store it into a list ']

3.

() Create a

new class and compare self : Name
with existing self - bar

Program NO: 39

Date: 21/21/2021

AIM: Python program to copy odd lines
of one file to another.

Program:

```
a = open('demo.txt', 'r')  
b = open('next demo.txt', 'w')  
c = a.readlines()
```

type(c)

for i in range(0, len(c)):

if (i%2 == 0):

b.write(c[i])

else:

pass

b.close()

b = open('next demo.txt', 'r')

cont1 = b.read()

print(cont1)

a.close()

b.close()

Next file: demo.txt

This is a python program to
go read a file line by line
and store it into a list

next demo.txt

To read a file line by line.

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

Output

To read a file line by line.

File to open in write

File to open in read mode

File : (compo1) base -> file

File (second) copy file

File (third) file

File (fourth) file

(file) copy file

(file.comb) base -> file

File (fifth) copy file

Copy file in each

Second and base of

File after the state func

File after the state func

This associ and compo1 : third

File after the state func

File after the state func

File after the state func

Program No: 40

Date: 21/2/2021

AIM: Write a python program to read each row from a given csv file and print a list of strings.

Program:

```
import csv
```

```
with open('departments.csv', newline='')
```

```
as csvfile:
```

```
data = csv.reader(csvfile, delimiter=',',
```

```
quotechar = '')
```

```
for row in data:
```

```
    print(','.join(row))
```

departments.csv:

department-id, department-name, Manager-id, location-id

10, Administration, 200, 1700

20, Marketing, 114, 1800

30, Human Resources, 203, 1700

40, Purchasing, 121, 2400

50, shipping, 103,1500

60, IT, 204, 1400

70, public Relation, 201, 2700

80, sales, 145, 2500.

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

Output

department-id, department-name, manager-id, location
10, Administration, 100, 1700
20, Marketing, 114, 1800
30, Human resources, 203, 1700
40, Purchasing, 121, 2400
50, Shipping, 103, 1500
60, IT, 204, 1400
70, Public relation, 201, 2700
80, Sales, 145, 2500

Program No: 4

Date: 20/2/2021

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

Program:

```
import csv  
with open('departments.csv', newline='')  
as csvfile:  
    data = csv.DictReader(csvfile)  
    print("ID Department name")  
    for row in data:  
        print(row['department-id'],  
              row['department-name'])
```

departments.csv :

department-id, department-name, manager-id, location-id

10, Administration, 200, 1700

20, Marketing, 201, 1800

30, Human Resource, 203, 2400

40, Purchasing, 114,1700

50, Shipping, 121,1500

60, IT, 103,1400

70, Public Relation, 204,12700

80, Sales, 145,2500.

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

Output

- 10 Department name
- 10 Administration
- 20 Marketing
- 30 Human Resources
- 40 Purchasing
- 50 Shipping
- 60 IT
- 70 Public Relations
- 80 Sales.

Date: 21/2/2021

Program NO: 42

AIM: write a python program to write a python dictionary to a csv file.

After writing the csv file read the csv file and display the content.

Program:

```
import csv
```

```
CSV_columns = ['id', 'column1', 'column2',  
               'column3']
```

```
dict data = {'id': ['1', '2', '3'],  
            'column1': [33, 25, 56],  
            'column2': [35, 30, 30],  
            'column3': [21, 40, 55]}
```

```
CSV_file = "temp.csv"
```

try:

```
    with open(CSV_file, 'w') as csvfile:  
        writer = csv.DictWriter(csvfile,  
                               fieldnames=CSV_columns)
```

```
writer.writerow()
```

For data in dictdata:

```
writer.writerow(dictdata)
```

except IOError:

```
print ("I/O Error")
```

```
data = csv.DictReader(open('CSV-File'))
```

```
.Print ("CSV file dictionary: \n")
```

```
for row in data
```

```
print (row)
```

Department.csv:

department_id, department_name, manager_id, location_id

10 , Administration, 200, 1700

20 , Marketing , 201, 1800

30 , Human Resources, 203, 2400

40 , Purchasing , 114, 1700

50 , Shipping, 121, 1500

60 , IT, 103, 1400

70, Public Relation, 204, 2700

80, Sales, 145, 2500

Temp.csv:

id, column1, column2, column3

"[1,2,3]", "[33,25,56]", "[35,30,30]", "[21,40,55]"

"[1,2,3]", "[33,25,56]", "[35,30,30]", "[21,40,55]"

"[1,2,3]", "[33,25,56]", "[35,30,30]", "[21,40,55]"

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

Output

csv file dictionary:

```
{'id': "[1,'2','3']",'column1': [33,25,56],  
'column2': [35,30,30],'column3': [21,40,55]
```

```
{'id': "[1,'2','3']",'column1': [33,25,56],  
'column2': [35,30,30],'column3': [21,40,55]
```

```
{'id': "[1,'2','3']",'column1': [33,25,56],  
'column2': [35,30,30],'column3': [21,40,55]
```

```
{'id': "[1,'2','3']",'column1': [33,25,56],  
'column2': [35,30,30],'column3': [21,40,55]
```