Date : 12-01-2021

program NO : 1

AIM : Program to check if else condition.

```
a = 12
b = 17
if (a > b):
    print(" the value of a is greater than b")
else:
    print(" the value of b is greater than a")
```

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

output

the value of b is greater than a.

Date : 16-01-2021

Program NO : 2

AIM : Program to find the square of a number entered by user.

```
num = int(input("enter a number: "))
square = num * num
print(f"square of {num} is {square}")
```

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

Output

Enter a number 9

Square of 9 is 81

Date : 16-01-2021

Program No : 3

AIM : Program to find area of a circle using a function.

```
def circleArea(r):
    PI = 3.14
    return PI * (r*r)

n = float(input("enter the value"))
print("area is %.6f" % circleArea(n))
```

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

Output

enter r value : 7

area is 153.860000

Date : 16-01-2021

Program No : 4

AIM : Program to find the biggest of three numbers entered by the user.

```
num1 = float(input("enter a number:"))
num2 = float(input("enter a number:"))
num3 = float(input("enter a number:"))

If (num1 > num2) and (num1 > num3):
        largest = num1

elif (num2 > num1) and (num2 > num3):
        largest = num3

Print("The largest number is", largest).
```

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

Output

enter a number : 3
enter a number : 5
enter a number : 8
the largest number is 8.

Date : 27-01-2021

Program NO : 5

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

```
def word(sta):
    counts = dict()
    words = sta.split()
    for i in words:
        if i in counts:
            counts[i] += 1
        else:
            counts[i] = 1
    return counts
print(word('have a nice day'))
```

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

output

{'have': 1, 'a': 1, 'nice': 1 'day': 1}

Date : 27-01-2021

Program No : 5

AIM : Store a list of first names, count the occurrences of 'a' within the list

```
newline = "say hai anna"
count = 0
for i in newline:
    if i == 'a':
        count = count + 1
print("count of a in say hai anna : " + str(count))
```

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

Output-

count of a in Say hai anna : 4

Date: 27-01-2021

Program No: 7

AIM: Get a string from an input string where all occurrences of first character replaced with '$' except first character.

```
def change_char(str1):
 char = str1[0]
str1 = str1.replace(char, '$')
str1 = char + str1[1:]
 return str1

print(change_char('restart'))
```

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

Output

restart

Program No: 8

Date: 27-01-2021

AIM: Create a string from given string where first and last characters exchanged.

```
def change_string(str1):
    return str1[-1:] + str1[1:-1] + str1[:1]
print(change_string('abcd'))
```

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

Output

dbca

Date : 27-01-2021

Program No : 9

AIM : Accept an integer n and compute n+nn+nnn.

```
n = int(input("enter a number"))
tp = str(n)
t1 = tp + tp
t2 = tp + tp + tp
comp = n + int(t1) + int(t2)
print("value", comp)
```

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

Output

Enter a number : 4

Value 999

Date: 27-01-2021

Program NO: 10

AIM: Merge two dictionaries

```
dict1 = {'a': 10, 'b': 8}
dict2 = {'d': 6, 'c': 4}
def merge(dict1, dict2):
    return(dict2.update(dict1))
print(merge(dict1, dict2))
print(dict2)
```

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

Output

{'d': 6, 'c': 4, 'a': 10, 'b': 8}

Date : 03-02-2021

Program No : 14

AIM : program to find the factorial of a number

```
num = int(input("enter a number"))
factorial = 1
If num < 0:
        Print("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 = factorial * i
print("The factorial of", num "is", factorial)
```

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

Output

Enter a number : 7

The factorial of 7 is 5040.

Date: 03-02-2021

Program No: 12

AIM: Generate fibonacci series of N terms

```
n = int(input("enter the value of 'n': "))
N = int(input("enter the
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 the output was verified.

Output

Enter the value of n : 5

0 1 1 2 3

Date : 03-02-2021

Program No : 13

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

```
sum = 0
li = [12, 134, 5, 4, 21]
for val in range[0, len(li)):
        sum = sum + li[val]
Print ("sum of all values in a given list",
        sum)
```

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

Output-

Sum of all values in a given list : 76

Date: 03-02-2021

Program No : 14

AIM : Count the number of characters in a string.

```
def char_freq(str1):
    dict = {}
    for n in str1:
        keys = dict.keys()
        If n in keys:
            dict[n] += 1
        else:
            dict[n] = 1
    return dict
print( char_freq('pathanamthitta'))
```

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

Output

{'p':1, 'a':4, 't':4, 'h':2, 'n':1, 'm':1
'i':1}

Date: 03-02-2021

Program No: 15

AIM: Adding 'ly' at the end of a given string. If it already ends with 'ing' then add 'ly'

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

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

Output

playing

playingly

Date : 03-02-2021

Program No : 16

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

```
def longest_word(words_li):
    word_len = []
    for n in words_li:
        word_len.append((len(n), n))
        word_len.sort()
    return word_len[-1][0], word_len[-1][1]
result = longest_word(["rose", "Jasmine", "lotus"])
print("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 : Jasmine

Length of the longest word : 7

Date : 03-02-2021

Program No : 17

AIM : Generate all factors of a number.

```
n = int(input("enter n"))
factors = [1]
for i in range(2, n):
    if(n % i == 0):
        factors.append(i)
        factors.append(n)
print(factors)
```

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

Output:

enter n 4

[1, 2, 4]

Date : 03-02-2021

Program No : 18

AIM : Lambda functions to find area of a square, rectangle and triangle.

```
s = int(input("enter the sides : "))
x = lambda a : a*a
print(x(s))
l = int(input("enter length : "))
b = int(input("enter breadth : "))
y = lambda l, b : l*b
print(y(l, b))
h = int(input("enter the height of triangle : "))
b1 = int(input("enter the base of triangle : "))
z = lambda h, b1 : (h*b1)/2
print(z(h, b1))
```

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

output

enter the sides : 4

16

enter the length : 6

enter breadth : 3

18

enter height of triangle : 7

enter base of triangle : 2

6.0

Date : 03-02-2021

Program No : 19

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

```
num = [5, 7, 20, 11, 18, 25, 4]
num = [x for x in num if x%2!=0]
print(num)
```

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

Output

[5, 7, 11, 25]

Date: 03-02-2021

Program No: 29

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

N = 4

```
1
2 4
3 6 9
4 8 12 16
```

```
def num():
    n = int(input("enter the number:"))
    i = 1
    for i in range(1, n+1):
        j = 1
    for j in range(1, i+1):
            temp = i * j
        print(temp, end = " ")
        print(" ")
num()
```

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

Output

```
Enter the number : 4
1
2 4
3 6 9
4 8 12 16
```

Date : 03-02-2021

Program No : 21

AIM : Construct following pattern using nested loop.

```
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 the output was verified.

Output

Program NO : 22

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

```
def gen():
    n=0
for k in range(1000, 10000):
    num = str(k)
    values = int(k)
    number1 = int(num[0])
    number2 = int(num[1])
    number3 = int(num[2])
    number4 = int(num[3])
    if number1 % 2 == 0:
        if number2 % 2 == 0:
            if number3 % 2 == 0:
                if number4 % 2 == 0:
                    for i in range(2, values):
                        if i*i == values:
                            print(values)
```

gen()

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

Output

4624
6084
6400
8464

Program No : 23

AIM : square of N numbers

```
def squares(n):
    L = [i*i for i in range(1, n+1)]
    return L

print(squares(15))
```

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

Output

[1, 4, 9, 16, 25, 36, 49, 64, 81, 100

121, 144, 196, 225]

Program NO : 24

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

```
word = input('enter a word')
v = {'a', 'e', 'i', 'o', 'u']
list = [ ]
for x in word:
    if (x in v and x not in list):
        list.append(x)
print("vowels that present", list)
```

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

Output

enter a word apple

vowels present ['a', 'e']

Program No : 25

AIM : Enter 2 lists of integers. check
1) whether list are of same length
2) whether list sums to same value
3) whether any value occurs in both.

```
def lists():
    list1 = []
    list2 = []
    list3 = []
    n1 = int(input("total number of elements in
        list1:"))
    for i in range(n1):
        val = int(input("enter the a number"))
        list1.append(val)
    n2 = int(input("total number of elements
        in the list2"))
```

```
for i in range(n2):
    val = int(input("enter a number"))
    list2.append(val)
if(n1==n2):
    print("list are of same length")
else:
    print("list are not same length")
if(sum(list1)==sum(list2)):
    print("sum value is same")
else:
    print("sum value is not same")
list3 = [each for each in list1 if each in
        list2]
print("values in the both lists are:" list3)

list()
```

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

Output

total number of elements in list1 : 4

enter a number : 5

enter a number : 6

enter a number : 2

enter a number : 4

total number of elements in the list-2 : 5

enter a number : 2

enter a number : 5

enter a number : 7

enter a number : 8

enter a number : 9

list are not same length

Sum value is not same

values in the both lists are : [5, 2]

Program NO : 26

AIM : Accept the radius from user and find area of circle.

```
def cirarea(r):
    PI = 3.14
    return PI * (r*r).

num = float(input("enter r value"))
print ("area is %4f" % cirarea(num))
```

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

Output

enter r value : 7

area is 153.860000

Program No : 27

AIM : Sort dictionary in ascending and descending order.

```
import operator
dt = {1:5, 3:4, 4,4:3, 2:1, 0:0}
print('dictionary':', dt)
S = dict(sorted(dt.items(), key = operator.
itemgetter(1)))
print('ascending order:', s)
S1 = dict(sorted(dt.items(), key = operator.
    itemgetter(1), reverse = True))
print('descending order:', s1)
```

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

Output

dictionary : {1:5, 3:4, 4:3, 2:1, 0:0}

ascending order : {0:0, 2:1, 4:3, 3:4, 1:5}

descending order: {1:5, 3:4, 4:3, 3:1, 0:0}

Program NO: 28

AIM: Find gcd of 2 numbers.

```
import math
print("the gcd of 60 and 40 is :",
      math.gcd(60,40))
```

Result: The program has been executed

Output-

the gcd of 60 and 40 is : 20

program NO: 29

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

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

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

Output

Enter final year : 2035

Leap years:

2024
2028
2032

Program No : 30

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

```
colors = input('enter colors : ').split(',')
print('first color :', colors[0])
print('Last color :', colors[len(colors)-1])
```

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

Output

Enter colors : red, black, rose, white

First color : red

Last color : white

Program No : 31

AIM : Print out all colors from color_list not contained in color_list2.

```
colors1 = set((input("Enter colors separated by commas : ")).split(','))

colors2 = set((input("Enter colors separated by commas : ")).split(','))

print('colors in color-list-1 not contained in color-list2 are :', list(colors1.difference(colors2)))
```

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

Output

Enter colors separated by commas: red, green, yellow

Enter colors separated by commas: blue, yellow, black

Colors in color_list1 not contained in color_list2 are ['green', 'red']

Program NO : 32

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

```
str1 = input('enter a string:')
str2 = input('enter another string :')
str3 = str2[0] + str1[1:] + ' ' + str1[0] + str2[1:]
print(str3)
```

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

Output

Enter a string : are you ok now

Enter another string : come let us play

are you ok now come let us play.

Program NO : 33

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

```
list1 = [11, -24, 0, 45, 66, -93]
for num in list1:
    if num >=0:
print(num, end = " ")
```

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

Output

11 0 45 66

Program No : 34

AIM : Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare two rectange object by their area.

```
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 comp(self, obj):
        if self.area() > obj.area():
            Print('Rectangle with length =',
                  self.length,' and breadth
```

```
=', self.breadth, 'has the greater area')
    elif self.area() < obj.area():
        print('Rectangle with length =', obj.length
and breadth =', obj.breadth, 'has the
greater area')
else:
    print('they have equal area!')
r1 = Rectangle(8, 2)
r2 = Rectangle(5, 7)
r1.comp(r2)
```

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

Output

Rectangle with length = 5 and breadth = 7 has the greater area.

Program NO : 35

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

```
Class BankAccount:
 def __init__(self, a, n, 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:
        self.bal -= a
        print ('Rs.', a, 'withdraw ! current balance is
                Rs ', self.bal)

    else:
        print ('Insufficient balace 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'))
acl = Bank Account(a, n, t, b)
acl.deposit( float(input('Enter amount to
        deposit :')))
acl.withdraw (float(input('Enter amount of
                withdraw :')))
```

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

Output

Enter account number : 19371554951

Enter name of the account holder : JDhn

Enter account type : savings

Enter your balance : 20000

Enter amount to deposit : 1000000

Rs. 1000000.0 deposited! current balance is : Rs. 1020000.0

Enter amount to withdraw! 300000

Rs. 300000.0 withdraw! current balance is Rs. 720000.0

program NO : 36

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

```
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 =', self.length,
'and width=', self.width, 'has the lesser
 area!')
elif other.area < self.area:
Print('Rectangle with length =',
```

```
other.length', and width = ', other.width',
has 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 width of 2nd rectangle'))
R2=Rectangle(l,w)
R1<R2
```

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

Output

Enter length of 1st rectangle : 7
Enter width of 1st rectangle : 4
Enter length of 2nd rectangle : 9
Enter width of 2nd rectangle : 3

Rectangle width length = 9.0 and width = 3.0 has the lesser area!

program NO : 37

AIM : Create a class Time with private attribute hour, minute and second. overload '+' Operator to find sum of 2 time.

```
class Time:
def __init__(self, hh, = 0, mm=0, ss =0):
  self.hour = hh
  self.minute = mm
  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) % 24 +
  self.minute + other.minute )/60)
Print('Time [hh:mm:ss]', hour':'minute':'
          second)
```

```
T1 = Time(13,24,45)
T2 = Time(8,52,46)
T1 + T2
```

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

Output

Time[hh:mm:ss] 22:17:31

program NO : 38

AIM: Create 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.

```
Class publisher:
  def __init__ (self, name1):
    self.name = name1
  def show(self):
  Pass
Class Book (publisher):
  def __init__ (self, title1, author1, name1):
  self.title = title1
  self.author = author1
```

```
publisher.__init__(self, name1)
def show(self):
    pass
class python(Book):
def __init__(self, p, no, title1, author1, name1):
self.price = p
self.no_of_pages = no
Book.__init__(self, title1, author1, name1)
def show(self):
print('Book title', self.title)
print('Author', self.author)
print('publisher', self.name)
print('price : Rs', self.price)
print('No of pages', self.no_of_pages)
P1 = Python(500, 544, 'learning python',
            'mark Lutz', 'ABC Books')
P1.show()
```

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

output

Book title : learning Python

Author : Mark Lutz

Published : ABC Books

price : Rs, 500

No : of pages : 544

Program No : 39

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

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

file_read("demo.text")
```

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

Program No : 40

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

```
a = open ('demo.txt', 'r')
b = open ('t.txt', 'w')
c = a.readlines()
for i in range (0, len(c)):
If (i.2 != 0):
b.write (c[i])
else:
 pass
b.close()
d = open ('t.txt', 'r')
```

```
d = b.read()
print(readl)
print(d)
a.close()
b.close()
```

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

Program No : 41

AIM : Write a Python program to read each row from a given csv files and print a list of strings

```
import csv
with open('cs.csv', newline = '') as csvfile:
d= csv.reader(csvfile, delimiter=',', quotechar
                = '|')
for r in d:
print(',', join(r))
```

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

Program No : 42

AIM: Write a python program to read specfic columns of a given csv files and print the content of the columns.

```
import csv
with open('c1.csv', newline='') as csvfile:
    d = csv.DictReader(csvfile)
    print("authors original_title")
    for r in d:
        print(r['authors'], r['original_title'])
```

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

program NO : 43

AIM: Write a python program to write a python directory to a csv file. After writing the csv file read the csv file and display the content.

```
import csv

field_names = ['best_book_id', 'authors',
               'original_title']

book = [
  {'best_book_id ": 1, 'authors': Suzanne
     Collins', 'original_title': 'The Hunger
     games' };

  {'best_book_id': 2, 'authors': 'J.K Rowling,
Mary Grandpre', 'original_title': 'The
```

philosophers stone'

{
'best_book_id': 3, 'authors':
'stephenine meyer', 'original_title':
'Twilight' };
]

```
with open('c#.csv', 'w') as csvfile:
writer = csv.DictWriter(csvfile, fieldnames=field
                                    -names)

writer.writeheader()
writer.writerows(book)

with open('c1'.csv', newline='') as csvfile:
d= csv.reader(csvfile, delimiter='|')
for r in d:
 Print(','.join(r))
```

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

Program No : 44

AIM: Python program to create a package graphic with modules Rectangle, circle, and sub package 3D-graphics including modules to find area and perimeter of respective figures in each module. Write program to find area and perimeter of figure by different importing statements.

circle.py

```
def area(r):
print("Area of circle with radius',r,' is:
    '%.2f' %.(3.14 * r * r), 'squnits')

def circumstance (r):
print ('circumstance of circle with radius.r
    'is:' '%.2f' %.(3.14*2*r), 'units')
```

rectangle.py

```
def area(a,b):
    print("area of rectangle with sides', 'and
    6 'is':'%.2f', % a*b, 'squnits')
def perimeter(a,b):
    print('Perimeter of rectangle with sides', a
    'and' 6, 'is:' '%.2f' % 2*(a+b)): 'units)
```

sphere.py

```
def area(r):
    print('Area of sphere with radius', r,
    'is:', '%.2f %.(4*(3.14*r*r)), 'sq.unit')
def perimeter(r):
    Print("perimeter of great circle of) sphere
    with radius', r, 'is' '%.2f', %.(2*3.12*r);
    'units')
```

cuboid.py

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

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

Find perimeter.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'))
perimeter(a, b)

r = float(input('enter the radius of the
                circle:'))
circle.circumstance(r)
l = float(input('enter length of the cuboid'))
b = float(input('enter breadth of the cuboid'))
r = float(input('enter the radius of the
          sphere:'))
sphere.perimeter(r)
```

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 the
                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)
```

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

Output

Enter length of the rectangle : 4
Enter breadth of the rectangle : 3
Perimeter of rectangle with sides 4.0 and 3.0 is : 14.00 unit

Enter the radius of the circle : 2
Circumstance of circle with radius 2.0 is 12.56 units

Enter length of the cuboid : 5
Enter the breadth of the cuboid : 4
Enter the height of the cuboid : 3
perimeter of the cuboid with dimension 4.0, 3.0 & 5.0 is 48.00 units

Enter the radius of the sphere is 2
Perimeter of (great of circle of) sphere with radius 2.0 is 12.56 units.

Enter length of the rectangle : 2
Enter breadth of the rectangle : 3

Area of rectangle with side 2.0 and 3.0 is
6.00 sq. units.

Enter the radius of the circle: 4
Area of circle with radius 4.0 is 50.24
sq. unit.

Enter length of the cuboid: 4
Enter breadth of the cuboid: 7 is
Enter height of the cuboid: 2
total surface area of cuboid with dimensions
4.0, 7.0, 2.0 is 100.00 sq. units

Enter the radius of the sphere
area of sphere with radius 1.0 is
12.56 sq. units.