Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming- - 2301CS404

Tanisha Bhalodiya - 23010101021

Lab - 1

01) WAP to print “Hello World”

In [1]:
```
print('hello world')
```

```
hello world
```

02) WAP to print addition of two numbers with and without using input().

In [1]:
```
num1=5
num2=9
print("Without input sum is : ",num1+num2);
a=int(input("enter a : "))
b=int(input('enter b : '))
c=a+b
print('sum is : ',c)
```

```
Without input sum is :  14
sum is :  11
sum is :  8
```

03) WAP to check the type of the variable.

In [7]:
```
var= int(input("enter variable : "))
print(type(var))
```

```
enter variable :  15
<class 'int'>
```

04) WAP to calculate simple interest.

In [9]:
```
p=float(input("Enter principle amount : "))
r=float(input("Enter rate of interest : "))
t=float(input("Enter time : "))
print("Simple interest is : ",(p*r*t)/100)
```

```
Enter principle amount :  1000
Enter rate of interest :  2
Enter time :  2
Simple interest is :  40.0
```

05) WAP to calculate area and perimeter of a circle.

In [11]:
```
rad=float(input("enter radius of circle: "))
print("area of circle : ",((3.14)*r*r))
```

```
enter radius of circle:  5
area of circle :  12.56
```

06) WAP to calculate area of a triangle.

In [13]:
```
base=float(input("enter base of triangle: "))
height=float(input("enter height of triangle: "))
print("area of triangle : ",((height*base)/2))
```

```
enter base of triangle:  20
enter height of triangle:  25
area of triangle :  250.0
```

07) WAP to compute quotient and remainder.

In [17]:
```
n1=float(input("Enter number 1: "))
n2=float(input("Enter number 2: "))
print("quotient is : ",(n1/n2))
print("remainder is : ",(n1%n2))
```

```
Enter number 1:  52
Enter number 2:  6
quotient is :  8.666666666666666
remainder is :  4.0
```

08) WAP to convert degree into Fahrenheit and vice versa.

In [19]:
```
cal=float(input("Enter calcius : "))
print("value in fehrenheit : ",((cal*(9/5))+32))

fah=float(input("Enter fahrenheit : "))
print("value in calcius : ",(fah-32)*(5/9))
```

```
Enter calcius :  40
value in fehrenheit :  104.0
Enter fahrenheit :  15
value in calcius :  -9.444444444444445
```

09) WAP to find the distance between two points in 2-D space.

In [27]:
```
import math
x1=float(input("enter x1 :" ))
y1=float(input("enter y1 :" ))
x2=float(input("enter x2 :" ))
```

```
y2=float(input("enter y2 :" ))
print("distance between two ponits : ", math.sqrt((x2 - x1)**2 + (y2 - y1)**2))
```

```
enter x1 : 2
enter y1 : 3
enter x2 : 5
enter y2 : 3
distance between two ponits :  3.0
```

10) WAP to print sum of n natural numbers.

In [45]:
```
n = int(input("Enter n: "))
sum = 0
for i in range(1, n + 1):
    sum =sum+i

print("Sum of first", n, "natural numbers is:", sum)
```

```
Enter n:  2
Sum of first 2 natural numbers is: 3
```

11) WAP to print sum of square of n natural numbers.

In [51]:
```
m = int(input("Enter m: "))
sum = 0
for i in range(1, m + 1):
    sum =sum+(i)*(i)

print("Sum of first", m, "natural numbers is:", sum)
```

```
Enter m:  3
Sum of first 3 natural numbers is: 14
```

12) WAP to concate the first and last name of the student.

In [61]:
```
fName=input("Enter first name : ")
lName=input("Enter last name : ")
full=fName + " " +lName
print("Full name is : ",full)
```

```
Enter first name :  tanisha
Enter last name :  patel
Full name is :  tanisha patel
```

13) WAP to swap two numbers.

In [69]:
```
print("with hepl of third variable : ")
l1=int(input("enter number 1: "))
l2=int(input("enter number 2: "))
l3=l1
l1=l2
l2=l3
print("number 1 is : ",l1,"number 2 is : ",l2)
print("without third variable: ")
l1=l1+l2 #2+5=7
l2=l1-l2 #7-5=2
```

```
l1=l1-l2 #7-2=5
print("number 1 is : ",l1,"number 2 is : ",l2)
```

```
with hepl of third variable :
enter number 1:  5
enter number 2:  2
number 1 is :  2 number 2 is :  5
without third variable:
number 1 is :  5 number 2 is :  2
```

14) WAP to get the distance from user into kilometer, and convert it into meter, feet, inches and centimeter.

In [73]:
```
km=float(input("enter distance in kilometers : "))
meter=km * 1000
feet = km * 3280.84
inches = km * 39370.1
centimeters = km * 100000
print(f"{meter} meters")
print(f"{feet} feet")
print(f"{inches} inches")
print(f"{centimeters} centimeters")
```

```
enter distance in kilometers :  20
20.0 kilometers is equal to:
20000.0 meters
65616.8 feet
787402.0 inches
2000000.0 centimeters
```

15) WAP to get day, month and year from the user and print the date in the given format: 23-11-2024.

In [79]:
```
day=int(input("enter days : "))
month=int(input("enter month : "))
year=int(input("enter year : "))
print(f"{day:02d}-{month:02d}-{year}")
print(day,"-",month,"-",year)
```

```
enter days :  23
enter month :  11
enter year :  2024
23-11-2024
23 - 11 - 2024
```

In [ ]:
```

```

Python Programming - 2301CS404

Tanisha Bhalodiya - 23010101021

Lab - 2

if..else..

01) WAP to check whether the given number is positive or negative.

In [5]:
```
n1=int(input("Enter number : "))
if(n1>0):
    print("Positive")
else:
    print("negative")
```

```
Enter number :  -3
negative
```

02) WAP to check whether the given number is odd or even.

In [9]:
```
n2=int(input("Enter number : "))
if(n2%2==0):
    print("number is Even")
else:
    print("number is odd")
```

```
Enter number :  3
number is odd
```

03) WAP to find out largest number from given two numbers using simple if and ternary operator.

In [19]:
```
a=int(input("Enter number a : "))
b=int(input("Enter number b : "))
if(a>b):
    print("a is largest")
```

```
else:
    print("b is largest")

max=print("A is bigger") if a>b else print("B is bigger")
```

```
Enter number a :  25
Enter number b :  20
a is largest
A is bigger
```

04) WAP to find out largest number from given three numbers.

In [29]:
```
x=int(input("Enter number x : "))
y=int(input("Enter number y : "))
z=int(input("Enter number z : "))
if(x>y):
    if(x>z):
        print("X is largest")
    else:
        print("Z is largest")
else:
    if(y>z):
        print("Y is largest")
    else:
        print("Z is largest")

max = (print("X is biggest") if x> z else print("Z is biggest")) if x>y else (pr
```

```
Enter number x :  1
Enter number y :  2
Enter number z :  3
Z is largest
Z is biggest
```

05) WAP to check whether the given year is leap year or not.

[If a year can be divisible by 4 but not divisible by 100 then it is leap year but if it is divisible by 400 then it is leap year]

In [39]:
```
year=int(input("Enter year : "))
if((year%4==0) and (year%100!=0)):
    print("Given year is not a leap year")
elif(year%400==0):
    print("Given year is leap year")
else:
    print("not a leap year")
```

```
Enter year :  2024
Given year is not a leap year
```

06) WAP in python to display the name of the day according to the number given by the user.

In [51]:
```
day=int(input("Enter day in number : "))
match day:
```

```
    case 1:print('Monday'),
    case 2:print('TuesDay'),
    case 3:print('Wednesday'),
    case 4:print('Thursday'),
    case 5:print('Friday'),
    case 6:print('Saturday'),
    case 7:print('Sunday')
```

```
Enter day in number :  7
Sunday
```

07) WAP to implement simple calculator which performs (add,sub,mul,div) of two no. based on user input.

In [71]:
```
num1=int(input("Enter number 1 : "))
num2=int(input("Enter number 2 : "))

op=input("Enter operation add,sub,mul and div : ")
match op:
    case '+':
        print("Addition is : ",int(num1+num2)),
    case '-':
        print("Subtraction is : ",int(num1-num2)),
    case '*':
        print("Multiplication is : ",num1*num2),
    case '/':
        print("Devision is : ",num1/num2),
```

```
Enter number 1 :  3
Enter number 2 :  2
Enter operation add,sub,mul and div :  +
Addition is :  5
```

08) WAP to read marks of five subjects. Calculate percentage and print class accordingly.

Fail below 35 Pass Class between 35 to 45 Second Class between 45 to 60 First Class between 60 to 70 Distinction if more than 70

In [73]:
```
s1=int(input("Enter mark od subject 1 : "))
s2=int(input("Enter mark od subject 2 : "))
s3=int(input("Enter mark od subject 3 : "))
s4=int(input("Enter mark od subject 4 : "))
s5=int(input("Enter mark od subject 5 : "))

per=((s1+s2+s3+s4+s5)/500)*100
if(per<35):
    print("Fail")
elif(per<45 and per>35):
    print("Pass calss")
elif(per>=45 and per<60):
    print("Second calss")
elif(per>=60 and per<70):
    print("First calss")
elif(per>=70):
    print("Distinction calss")
```

```
Enter mark od subject 1 :  50
Enter mark od subject 2 :  60
Enter mark od subject 3 :  70
Enter mark od subject 4 :  80
Enter mark od subject 5 :  90
Distinction calss
```

09) Three sides of a triangle are entered through the keyboard, WAP to check whether the triangle is isosceles, equilateral, scalene or right-angled triangle.

In [13]:
```
l1=float(input("Enter side 1 : "))
l2=float(input("Enter side 2 : "))
l3=float(input("Enter side 3 : "))

if((l1+l2>l3) and (l2+l3 >l1) and (l3+l1>l2)):
    if(l1==l2==l3):
        print("Equilateral trianle")
    elif(l1==l2 or l1==l3):
        print("isosceles triangle")
    elif((l1*l1 == l2*l2 + l3*l3) or (l2*l2 == l1*l1 + l3*l3) or (l3*l3 == l1*l1
        print("Right-angled triangle")
    else:
        print("scalene triangle")
else:
    print('This Triangle is not possible')
```

```
Enter side 1 :  3
Enter side 2 :  4
Enter side 3 :  7
This Triangle is not possible
```

In [ ]:
```

```

10) WAP to find the second largest number among three user input numbers.

In [5]:
```
x1=int(input("Enter number x1 : "))
y1=int(input("Enter number y1 : "))
z1=int(input("Enter number z1 : "))
arr=[x1,y1,z1]
arr.sort()
print("Second largest is : ",arr[1])
print("=======================================")
if(x1>y1 and x1>z1):
    if(y1>z1):
        print("Y1 is second largest")
    else:
        print("Z1 is second largest")
elif(y1>x1 and y1>z1):
    if(x1>z1):
        print("X1 is second largest")
    else:
        print("Z1 is second largest")
else:
    if(x1>y1):
        print("x1 is second largest")
```

```
    else:
        print("y1 is second largest")
```

```
Enter number x1 :  2
Enter number y1 :  3
Enter number z1 :  4
Second largest is :  3
=======================================
y1 is second largest
```

11) WAP to calculate electricity bill based on following criteria. Which takes the unit from the user.

a. First 1 to 50 units – Rs. 2.60/unit b. Next 50 to 100 units – Rs. 3.25/unit c. Next 100 to 200 units – Rs. 5.26/unit d. above 200 units – Rs. 8.45/unit

In [1]:
```
units=float(input("Enter units : "))
if units <=50:
    bill =units *2.60
elif units <=100:
    bill =(50*2.60)+((units -50)*3.25)
elif units <=200:
    bill =(50*2.60)+(50*3.25)+((units -100)*5.26)
else:
    bill =(50*2.60)+(50*3.25)+(100*5.26)+((units -200)*8.45)
print(f'Bill is : {bill}')
```

```
Bill is : 397.7
```

In [ ]:
```
```

Darshan UNIVERSITY
योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Tanisha Bhalodiya - 23010101021

Lab - 3

for and while loop

01) WAP to print 1 to 10.

In [3]:
```
for i in range(1,11):
    print(i)
```

```
1
2
3
4
5
6
7
8
9
10
```

02) WAP to print 1 to n.

In [7]:
```
n2=int(input("Enter number n2 : "))
for i in range(1,n2+1):
    print(i)
```

```
Enter number n2 :  2
1
2
```

03) WAP to print odd numbers between 1 to n.

In [25]:
```
n3=int(input("Enter number n3 : "))
for i in range(1,n3+1):
```

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

```
Enter number n3 :  6
1
3
5
```

04) WAP to print numbers between two given numbers which is divisible by 2 but not divisible by 3.

In [27]:
```
n4=int(input("Enter number n4 : "))
for i in range(1,n4+1):
    if(i % 2 ==0 and i%3 !=0):
        print(i)
```

```
Enter number n4 :  6
2
4
```

05) WAP to print sum of 1 to n numbers.

In [37]:
```
n5=int(input("Enter number n5 : "))
sum=0
for i in range(1,n5+1):
    sum=sum+i
print("sum os 1 to n is : ",sum)
```

```
Enter number n5 :  5
sum os 1 to n is :  15
```

06) WAP to print sum of series 1 + 4 + 9 + 16 + 25 + 36 + ...n.

In [4]:
```
n6=int(input("Enter number n6 : "))
sum2=0
for i in range(1,n6+1):
    print(i*i," + ",end=" ")
    sum2=sum2+(i*i)
print("sum is : ",sum2)
```

```
Enter number n6 :  5
1  +  4  +  9  +  16  +  25  +  sum is :  55
```

07) WAP to print sum of series 1 – 2 + 3 – 4 + 5 – 6 + 7 ... n.

In [71]:
```
n7=int(input("Enter number n7 : "))
sum3=0
count=0
for i in range(1,n7+1):
    if(count==0):
        sum3=sum3+i
        count=1
    else:
        sum3=sum3-i;
        count=0
print("sum is : ",sum3)
```

```
Enter number n7 :  4
sum is :  -2
```

08) WAP to print multiplication table of given number.

In [77]:
```
n=int(input("Enter number : "))
for i in range(1,11):
    print(f"{n} X {i} ={n*i}")
```

```
Enter number n7 :  2
2 X 1 =2
2 X 2 =4
2 X 3 =6
2 X 4 =8
2 X 5 =10
2 X 6 =12
2 X 7 =14
2 X 8 =16
2 X 9 =18
2 X 10 =20
```

09) WAP to find factorial of the given number.

In [83]:
```
n=int(input("enter number : "))
fact=1;
for i in range(1,n+1):
    fact=fact*i
print("Factorial of given number is : ",fact)
```

```
enter number :  5
Factorial of given number is :  120
```

10) WAP to find factors of the given number.

In [85]:
```
n10=int(input("Enter number for factors"))
print("Factorial of given number is : ")
for i in range(1,n10+1):
    if(n10 % i ==0):
        print(i)
```

```
Enter number for factors 6
Factorial of given number is :
1
2
3
6
```

11) WAP to find whether the given number is prime or not.

In [97]:
```
n11=int(input("Enter number for factors: "))
for i in range(2,int(n11/2)):
    if(n11%i==0):
        print("Number is not prime")
        break
else:
    print("Number is prime")
```

```
Enter number for factors:  6
Number is not prime
```

12) WAP to print sum of digits of given number.

In [110…

```
n12=int(input("Enter number: "))
sum4=0
while(n12>0):
    r=n12%10
    sum4=sum4+r
    n12=int(n12/10)
print("Sum of digits of given number is : ",sum4)
```

```
Enter number:  32
Sum of digits of given number is :  5
```

13) WAP to check whether the given number is palindrome or not

In [132…

```
n13=int(input("Enter number: "))
no=n13
sum5=0
while(n13>0):
    r=n13%10
    sum5=(sum5*10)+r
    n13=n13//10
if(sum5==no):
    print(f"{no} is palindrome")
else:
    print(f"{no} is not palindrome")
```

```
Enter number:  123321
123321 is palindrome
```

14) WAP to print GCD of given two numbers.

In [162…

```
import math
a=int(input("Enter number1: "))
b=int(input("Enter number2: "))
i=1
gcd=1
res=gcd
print("Gcd of numbers by math methood is : ",math.gcd(a,b))
while((a>0 or b>0) and (i<=a or i<=b)):
    if(a%i==0 and b%i==0):
        gcd=i
    if(res<=gcd):
        res=gcd
    i+=1
print(f"GCD of {a} and {b} is : ",res)
```

```
Enter number1:  36
Enter number2:  80
Gcd of numbers by math methood is :  4
GCD of 36 and 80 is :  4
```

In [ ]:

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Tanisha Bhalodiya - 23010101021

Lab - 4

String

01) WAP to check whether the given string is palindrome or not.

In [5]:
```
s1=input("Enter String : ")
if s1 == s1[::-1]:
    print("String is palindrome")
else:
    print("String is not palindrome")
```

```
String is palindrome
```

02) WAP to reverse the words in the given string.

In [21]:
```
s2=input("Enter String : ")
words=s2.split()
print("Without for loop reverse string is : ",s2[::-1])
str=''
for i in words:
    str=str+i[::-1]+' '
print("Reverse of given string is : ",s2)
```

```
Without for loop reverse string is :  ahsinat si eman ym
Reverse of given string is :  my name is tanisha
```

03) WAP to remove ith character from given string.

In [3]:
```
s3=input("Enter String : ")
i=int(input("Enter character you want to remove : "))
s3 = s3[:i:] + s3[i+1:]
print("String after removing : ",s3)
```

```
String after removing :  tanisha
```

04) WAP to find length of string without using len function.

In [21]:
```
s4=input("Enter String : ")
count=0
for i in s4:
    count=count+1
print("Length of string is : ",count)
```

```
Length of string is :  10
```

05) WAP to print even length word in string.

In [27]:
```
s5=input("Enter String : ")
words = s5.split()
print("Even length word in string : ")
for i in words:
    if len(i)%2==0:
        print(i)
```

```
Even length word in string :
hi
girl
```

06) WAP to count numbers of vowels in given string.

In [37]:
```
s6=input("Enter String : ").lower()
c=0
vowels='aeiou'
for i in s6:
    if i in vowels:
        c=c+1
print("Count numbers of vowels in given string is : ",c)
```

```
Count numbers of vowels in given string is :  3
```

07) WAP to capitalize the first and last character of each word in a string.

In [1]:
```
s7=input("Enter String : ")
word=s7.split()
j=0
s7=''
for i in word:
    j=len(i)-1
    i=i[0].capitalize()+i[1:j]+ i[j].upper()
    s7=s7+' '+i
print(s7)
```

```
TanishA PateL
```

08) WAP to convert given array to string.

In [134…
```
n=int(input("Enter size : "))
array=[]
string=''
for i in range(0,n):
```

```
    i=input("Enter ele : ")
    string=string+i
print("String is : ",string)
```

```
String is :  tanisha
```

09) Check if the password and confirm password is same or not.

In case of only case's mistake, show the error message.

In [64]:
```
pwd=input("Enter password : ")
cpwd=input("Enter confirm password : ")

if pwd==cpwd:
    print("Password is totally correct")
else:
    if pwd.lower()==cpwd.lower():
        print("Password is correct but case is different")
    else:
        print("unfortunatly Password is not same")
```

```
Password is correct but case is different
```

10) : Display credit card number.

card no. : 1234 5678 9012 3456

display as : **** **** **** 3456

In [68]:
```
str=input("Enter string : ")
print("card no. : ",str)
print("display as: **** **** ****",str[-4:])
```

```
card no. :  1234 5678 9012 3456
display as: **** **** **** 3456
```

11) : Checking if the two strings are Anagram or not.

s1 = decimal and s2 = medical are Anagram

In [3]:
```
str1=input("Enter string 1 : ")
str2=input("Enter string 2 : ")
c=0
# if sorted(str1)==sorted(str2):
#     print("Strings are anagram")
# else:
#     print("Strings are not anagram")

for i in str1:
    if i in str2:
        c=c+1
if c==len(str1):
    print("Strings are anagram")
```

```
else:
    print("Strings are not anagram")
```

```
Strings are anagram
```

12) : Rearrange the given string. First lowercase then uppercase alphabets. input : EHlsarwiwhtwMV output : lsarwiwhtwEHMV

In [104…

```
string=input("Enter String : ")
lower=''
upper=''
for i in string:
    if i.islower():
        lower=lower+i
    else:
        upper=upper+i
string=lower+upper
print("String is : ",string)
```

```
String is :  lsarwiwhtwEHMV
```

In [98]:

In [ ]:

Darshan UNIVERSITY
योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Tanisha Bhalodiya - 23010101021

Lab - 5

List

01) WAP to find sum of all the elements in a List.

In [3]:
```
l1=[1,2,3,4,5]
sum=0
for i in l1:
    sum=sum+i
print(f'Sum is : {sum}')
```

```
Sum is : 15
```

02) WAP to find largest element in a List.

In [1]:
```
l2=[10,100,20,65,84]
largest=max(l2)
print('largset number is : ',largest)
```

```
largset number is :  100
```

03) WAP to find the length of a List.

In [15]:
```
l3=[1,2,3,4,5,6,7,8,9]
print(f'Length of list is : ',len(l3))
```

```
Length of list is :  9
```

04) WAP to interchange first and last elements in a list.

In [19]:
```
l4=[1,2,3,4,5]
last=len(l4)-1
l4[0] , l4[last] = l4[last] , l4[0]
```

```
print('After swap list is : ')
print(l4)
```

```
After swap list is :
[5, 2, 3, 4, 1]
```

05) WAP to split the List into two parts and append the first part to the end.

In [23]:
```
l=[1,2,3,4,5,6,7,8,9]
mid=len(l)//2
list1=l[:mid:]
list2=l[mid::]
newList=list2+list1
print("New list : ",newList)
```

```
New list :  [5, 6, 7, 8, 9, 1, 2, 3, 4]
```

06) WAP to interchange the elements on two positions entered by a user.

In [28]:
```
first=int(input("Enter first position : "))
second=int(input("Enter second position : "))
l[first],l[second]=l[second],l[first]
print("Changed array is : ",l)
```

```
Changed array is :  [1, 2, 7, 4, 5, 6, 3, 8, 9]
```

07) WAP to reverse the list entered by user.

In [32]:
```
list=[1,2,3,4,5,6,7]
list=list[::-1]
print('Reversed list is : ',list)
```

```
Reversed list is :  [7, 6, 5, 4, 3, 2, 1]
```

08) WAP to print even numbers in a list.

In [36]:
```
l8=[1,3,2,5,4,7,6,8]
print('Even number in list : ')
for i in l8:
    if i%2==0:
        print(i)
```

```
Even number in list :
2
4
6
8
```

09) WAP to count unique items in a list.

In [67]:
```
l9=[1,2,3,2,8,5,4,5,0,7,8,9]
l=[]
for i in l9:
    if i not in l:
```

```
        l.append(i)
print(len(l))
```

```
9
```

10) WAP to copy a list.

In [69]:
```
l10=[56,51,21,48,75,14]
list=l10.copy()
print(f'Copy list is : {list}')
```

```
Copy list is : [56, 51, 21, 48, 75, 14]
```

11) WAP to print all odd numbers in a given range.

In [2]:
```
n1=int(input('Enter first number for range : '))
n2=int(input('Enter last number for range : '))
for i in range(n1,n2+1):
    if i%2!=0:
        print(i)
```

```
3
5
7
9
```

12) WAP to count occurrences of an element in a list.

In [83]:
```
l9=[1,2,3,2,8,5,4,5,0,7,8,9]
occur=0
for i in l9:
    occur=l9.count(i)
    print(occur)
    l9.remove(i)
```

```
1
1
2
1
1
1
```

13) WAP to find second largest number in a list.

In [85]:
```
l10=[56,51,21,48,75,14]
lar=max(l10)
l10.remove(lar)
lar=max(l10)
print('second largest number is : ',lar)
```

```
second largest number is :  56
```

14) WAP to extract elements with frequency greater than K.

In [97]:
```
k=int(input('Enter k : '))
l9=[1,2,3,2,8,5,4,5,0,7,8,9,9,9,2,2,3,3,3,3]
fer=0
```

```
for i in l9:
    fer=l9.count(i)
    if fer>k:
        print(i)
    while i in l9:
        l9.remove(i)
```

```
3
9
2
```

15) WAP to create a list of squared numbers from 0 to 9 with and without using List Comprehension.

In [107…

```
print('Without comprehendtion : ')
list=[]
for i in range(0,10):
    list.append(i*i)
print(list)

print('With comprehension : ')
list1=[i*i for i in range(0,10)]
print(list1)
```

```
Without comprehendtion :
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
With comprehension :
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

16) WAP to create a new list (fruit whose name starts with 'b') from the list of fruits given by user.

In [111…

```
n=int(input('enter size of list : '))
fruits=[]
new=[]
for i in range(0,n):
    i=input('enter fruit')
    fruits.append(i)
    if i[0]=='b':
        new.append(i)
print(fruits)
print(new)
```

```
['banana', 'mango', 'blue berry']
['banana', 'blue berry']
```

17) WAP to create a list of common elements from given two lists.

In [117…

```
l1=[1,2,3,4,5,6]
l2=[5,6,7,8,9]
for i in l1:
    if i in l2:
        print(i)
```

```
5
6
```

In [ ]:

Darshan UNIVERSITY
योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Tanisha Bhalodiya - 23010101021

Lab - 6

Tuple

01) WAP to find sum of tuple elements.

In [2]:
```
t1=(1,2,3,4,5)
sum=0
for i in t1:
    sum += i
print(sum)
```

```
15
```

02) WAP to find Maximum and Minimum K elements in a given tuple.

In [8]:
```
t2=(3,1,2,6,7,0,9,10,35,34)
li=list(t2)
li.sort()
k=int(input("Enter k : "))
maxlist=li[:k:]
minlist=li[:-k-1:-1]
print(f'Maximum element is : {maxlist} and minimum element is : {minlist}')
```

```
Maximum element is : [0, 1, 2] and minimum element is : [35, 34, 10]
```

03) WAP to find tuples which have all elements divisible by K from a list of tuples.

In [89]:
```
li=[(1,2,3),(32,4,6),(1,3,5),(50,45,84,75),(2,4,6)]
k=int(input("Enter k : "))
for i in li:
    count=0
    for j in i:
```

```
        if(j%k==0):
            continue
        else:
            count=1
    if(count==0):
        print(i)
    # if(count=0):
    #     ans=tuple(j for j in i if j%k==0 else count=1)
    #     print(ans)
```

```
(32, 4, 6)
(2, 4, 6)
```

04) WAP to create a list of tuples from given list having number and its cube in each tuple.

In [100…

```
l4=[1,2,3,4,5,6,7,8,9]
result=[]
for i in l4:
    t=(i,i**2)
    result.append(t)
print(result)
```

```
[(1, 1), (2, 4), (3, 9), (4, 16), (5, 25), (6, 36), (7, 49), (8, 64), (9, 81)]
```

05) WAP to find tuples with all positive elements from the given list of tuples.

In [104…

```
l5=[(1,2,3),(-1,0,-9),(4,65,78),(0,2,8),(-9,-36,-98)]
print("Tuples with all positive elements are : ")
for t in l5:
    count=0
    for j in t:
        if(j>=0):
            continue
        else:
            count=1
    if(count==0):
        print(t)
```

```
(1, 2, 3)
(4, 65, 78)
(0, 2, 8)
```

06) WAP to add tuple to list and vice – versa.

In [118…

```
l6=[1,2,3,4,5]
t6=(5,6,7,8,9)

tuple_to_list=l6+list(t6)
list_to_tuple=t6+tuple(l6)

print('ADD TUPLE TO LIST : ',tuple_to_list)
print('ADD LIST TO TUPLE : ',list_to_tuple)
```

```
ADD TUPLE TO LIST :  [1, 2, 3, 4, 5, 5, 6, 7, 8, 9]
ADD LIST TO TUPLE :  (5, 6, 7, 8, 9, 1, 2, 3, 4, 5)
```

07) WAP to remove tuples of length K.

In [153…
```
l7=[(1,2),(1,1),(4,65,78),(2,8),(9,-36,-98,0),(1,)]
k=int(input('Enter k : '))
j=0
li=0
while(j<len(l7)):
    if(len(l7[j])==k):
        l7.pop(j)
        j=j-1
    j=j+1
print(l7)
```

```
[(1, 2), (1, 1), (2, 8), (9, -36, -98, 0), (1,)]
```

08) WAP to remove duplicates from tuple.

In [165…
```
tup=(1,2,3,2,4,5,6,5,7,8,1,0,2,4,4)
s=set(tup)
tup=tuple(s)
print('after removing duplicates from tuple: ')
print(tup)
```

```
after removing duplicates from tuple:
(0, 1, 2, 3, 4, 5, 6, 7, 8)
```

09) WAP to multiply adjacent elements of a tuple and print that resultant tuple.

In [189…
```
t9=(1,2,3,4,5,6)
t9=list(t9)
print(tuple(t9))
res=[]
print('multiply adjacent elements of a tuple : ')
for i in range(1,len(li)):
    res.append(t9[i-1]*t9[i])
print(tuple(res))
```

```
(1, 2, 3, 4, 5, 6)
multiply adjacent elements of a tuple :
(2, 6, 12, 20, 30)
```

10) WAP to test if the given tuple is distinct or not.

In [198…
```
t9=(1,2,3,4,5,6)
tup=(1,2,3,2,4,5,6,5,7,8,1,0,2,4,4)
if(len(tup)==len(set(tup))):
    print('tuple is distinct')
else:
    print('tuple is not distinct')
```

```
tuple is not distinct
```

In [4]:
```
lst = ["a", "b"]
lst.extend("AK")
lst.extend(["AK","AK"])
print(lst)
```

```
['a', 'b', 'A', 'K', 'AK', 'AK']
```

In [6]:
```
lst = [1, 2, 3]
print("index(-1) = ",lst[-1])
lst.insert(-1, "near end")
print(lst)  # [1, 2, 'near end', 3]
```

```
index(-1) =  3
[1, 2, 'near end', 3]
```

In [ ]:

In [ ]:

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Tanisha Bhalodiya - 23010101021

Lab - 7

Set & Dictionary

01) WAP to iterate over a set.

In [1]:
```
s1={1,2,3,4,5,6,7}
for i in s1:
    print(i)
```

```
1
2
3
4
5
6
7
```

2) WAP to convert set into list, string and tuple.

In [20]:
```
set1={1,2,3,4,5}
print('set is : ',set1)
list1=list(set1)
print('list is : ',list1)
tup1=tuple(set1)
print('Tuple is : ',tup1)
str1=''
for i in set1:
    str1=str1+str(i)
print('string is : ',str1)
```

```
set is :  {1, 2, 3, 4, 5}
list is :  [1, 2, 3, 4, 5]
Tuple is :  (1, 2, 3, 4, 5)
string is :  12345
```

03) WAP to find Maximum and Minimum from a set.

In [3]:
```
set1={1,2,3,4,5,6}
maximum=0
minimum=set1.pop()
set1.add(minimum)
for i in set1:
    if(i>maximum):
        maximum=i
    if(i<minimum):
        minimum=i

print('maximum is : ',maximum)
print('minimum is : ',minimum)
print('max is : ',max(set1))
print('min is : ',min(set1))
```

```
maximum is :  6
minimum is :  1
max is :  6
min is :  1
```

04) WAP to perform union of two sets.

In [1]:
```
s1={1,2,3,4}
s2={3,4,5,6,7,8}
print('union of two sets is : ',s1.union(s2))
```

```
union of two sets is :  {1, 2, 3, 4, 5, 6, 7, 8}
```

05) WAP to check if two lists have at-least one element common.

In [11]:
```
s1={1,2,3,4}
s2={3,4,5,6,7,8}
c=len(s1.intersection(s2))
if(c>=1):
    print('Sets have at-least one element common')
```

```
Sets have at-least one element common
```

06) WAP to remove duplicates from list.

In [9]:
```
list1=[1,2,3,4,4,5,5,6,6]
set2=list(set(list1))
print(set2)
```

```
[1, 2, 3, 4, 5, 6]
```

07) WAP to find unique words in the given string.

In [7]:
```
str2='this is python, python is very easy'
l7=str2.split()
set3=set(l7)
print(set3)
```

```
{'is', 'python,', 'very', 'python', 'easy', 'this'}
```

In [17]:
```
l1=[1,2,3]
l2=[10,10,10]
ans=list(map(lambda x,y:max(x,y),l1,l2))
print(ans)
```

```
[10, 10, 10]
```

08) WAP to remove common elements of set A & B from set A.

In [40]:
```
seta={1,2,3,4,5}
print('Set A is : ',seta)
setb={4,5,6,7,8}
print('Set B is : ',setb)
common=list(seta.intersection(setb))
print('Common element are : ',common)
for i in common:
    seta.remove(i)
print('After removing : ',seta)
```

```
Set A is :  {1, 2, 3, 4, 5}
Set B is :  {4, 5, 6, 7, 8}
Common element are :  [4, 5]
After removing :  {1, 2, 3}
```

09) WAP to check whether two given strings are anagram or not using set.

In [47]:
```
str3=input('Enter string 1: ')
l3=list(str3)
str4=input('Enter string 2: ')
l4=list(str4)
flag=False
for i in l3:
    if(i not in l4):
        flag=True
        print('Strings are not anagram')
if(flag==False):
    print('Strings are anagram')
```

```
Strings are anagram
```

In [33]:
```
from functools import reduce
s1=['a','b','c','d']
ans=reduce(lambda x,y:x+y,s1)
print(ans)
```

```
abcd
```

10) WAP to find common elements in three lists using set.

In [29]:
```
s1=[1,2,3,4,5]
s2=[4,5,6,7,8]
s3=[4,5,9,10,11]
l1=set(s1)
l2=set(s2)
```

```
l3=set(s3)
ele=l3.intersection(l2.intersection(l1))
print('Common elements : ',ele)
```

```
Common elements :  {4, 5}
```

11) WAP to count number of vowels in given string using set.

In [21]:
```
str5=input('Enter string with vowels : ')
vowel=set('aAeEiIoOuU')
count=0
for i in str5:
    if i in vowel:
        count+=1
print('Number of vowels in given string is : ',count)
```

```
Number of vowels in given string is :  3
```

In [51]:
```
l3=[11,22,33,44,11,22,36,14,11,11,11,22]
from functools import reduce
ans=list(filter(lambda x:l3.count(x) >= 2 and x%11==0,l3))
print(ans)
```

```
[11, 22, 11, 22, 11, 11, 11, 22]
```

12) WAP to check if a given string is binary string or not.

In [25]:
```
str5=input('Enter string : ')
binary=set('01')
flag=False
for i in str5:
    if i not in binary:
        flag=True
        print('Given string is not binary string')
if flag==False:
    print('Given string is binary')
```

```
Given string is not binary string
```

13) WAP to sort dictionary by key or value.

In [111…
```
dic={'one':1,'zero':0,'two':2,'four':4,'three':3,'five':5}
key=list(dic.keys())
value=list(dic.values())


for i in range(len(key)):
    for j in range(i+1,len(key)):
        if key[i]>key[j]:
            key[i],key[j]=key[j],key[i]
            value[i],value[j]=value[j],value[i]

sorted_by_key=dict(zip(key,value))
print('Sorted by keys : ',sorted_by_key)

for i in range(len(value)):
    for j in range(i+1,len(value)):
```

```
        if value[i]>value[j]:
            value[i],value[j]=value[j],value[i]
            key[i],key[j]=key[j],key[i]

sorted_by_value=dict(zip(key,value))
print('Sorted by values : ',sorted_by_value)

# we can do like this also..............

sort_by_key=dict(sorted(dic.items()))
print('Sorted by key : ',sort_by_key)
sort_by_value=dict(sorted(dic.items(),key=lambda item:item[1]))
print('Sorted by value : ',sort_by_value)

dic.items()
```

```
Sorted by keys :  {'five': 5, 'four': 4, 'one': 1, 'three': 3, 'two': 2, 'zero':
0}
Sorted by values :  {'zero': 0, 'one': 1, 'two': 2, 'three': 3, 'four': 4, 'fiv
e': 5}
Sorted by key :  {'five': 5, 'four': 4, 'one': 1, 'three': 3, 'two': 2, 'zero':
0}
Sorted by value :  {'zero': 0, 'one': 1, 'two': 2, 'three': 3, 'four': 4, 'five':
5}
```

Out[111… `dict_items([('one', 1), ('zero', 0), ('two', 2), ('four', 4), ('three', 3), ('f ive', 5)])`

14) WAP to find the sum of all items (values) in a dictionary given by user. (Assume: values are numeric)

In [59]:
```
dict1={}
sum1=0
n=int(input('Enter size of dictionary : '))
for _ in range(n):
    key=input('enter keys : ')
    value=int(input('enter value : '))
    dict1[key]=value
    sum1+=value
print('dictionary is : ',dict1)
print('Sum of values : ',sum1)
```

```
dictionary is :  {'one': 1, 'two': 2, 'three': 3}
Sum of values :  6
```

15) WAP to handle missing keys in dictionaries.

Example : Given, dict1 = {'a': 5, 'c': 8, 'e': 2}

if you look for key = 'd', the message given should be 'Key Not Found', otherwise print the value of 'd' in dict1.

In [115…
```
dict2={'a':1,'b':2,'c':3,'e':4,'h':5}
k=input('Enter key : ')
if(k in dict2.keys()):
    print('Values of key is : ',dict2.get(k))
else:
    print('KEY NOT FOUND')
```

```
KEY NOT FOUND
```

In [3]:
```python
# generate fibbonaci series without recursion

# n=int(input('Enter number : '))
# i=0
# j=1
# print(i)
# print(j)
# for c in range(n-2):
#     k=i+j
#     print(k)
#     i,j=j,k


# generate fibbonaci series with recursion
def fibbo(n):
    if n <= 1:
        return n
    else:
        return fibbo(n - 1) + fibbo(n - 2)

n=int(input('Enter number : '))
for i in range(n):
    print(fibbo(i))
```

```
0
1
1
2
3
```

In [ ]:
```python

```

Python Programming - 2301CS404

Tanisha Bhalodiya - 23010101021

Lab - 8

User Defined Function

01) Write a function to calculate BMI given mass and height. (BMI = mass/h**2)

In [30]:
```
def calculate_bmi(h, mass):
    bmi = mass / (h ** 2)
    return bmi
h=float(input('Enter height : '))
mass=float(input('Enter weight : '))
bmi=calculate_bmi(h,mass)
print('bmi is : ',bmi)
```

```
bmi is :  0.0021777777777777776
```

02) Write a function that add first n numbers.

In [38]:
```
n=int(input('Enter n : '))
def add_n(n):
    add=0
    for i in range(n):
        add=add+i
    print(f'Addition of first {n} numbers is {add} ')
add_n(n)
```

```
Addition of first 5 numbers is 10
```

03) Write a function that returns 1 if the given number is Prime or 0 otherwise.

In [58]:
```
num=int(input('Enter number : '))
def check_prime(num):
    i=2
```

```
    while(i<num):
        if(num%i==0):
            return 0
            break
        i+=1
    return 1
check_prime(num)
```

Out[58]: 1

04) Write a function that returns the list of Prime numbers between given two numbers.

In [78]:
```
n1=int(input('Enter number 1 : '))
n2=int(input('Enter number 2 : '))
prime=[]
def prime_list(n1,n2):
    for n in range(n1,n2):
        flag=0
        for i in range(2,n):
            if(n%i==0):
                flag=1
                break;
        if(flag==0):
            prime.append(n)
prime_list(n1,n2)
print(prime)
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47]
```

05) Write a function that returns True if the given string is Palindrome or False otherwise.

In [80]:
```
s1=input("Enter String : ")
def check_palindrome(s1):
    if s1==s1[::-1]:
        return True
    else:
        return False
check_palindrome(s1)
```

Out[80]: True

06) Write a function that returns the sum of all the elements of the list.

In [96]:
```
numbers=[1,2,3,4,5,6]
def sum_of_list(numbers):
    # we can use LIST COMPREHENTION
    total=sum([i for i in numbers])
    # total=0
    # for i in numbers:
    #     total+=i
    print('Sum of elements : ',total)
sum_of_list(numbers)
```

```
Sum of elements :  21
```

07) Write a function to calculate the sum of the first element of each tuples inside the list.

In [100…
```
l1=[(1,2),(2,3),(3,4),(4,5),(5,6)]
def sum_of_tuple(l1):
    sum1=0
    for i in l1:
        a,b=i
        sum1=sum1+a
    print('Sum of first element in tuple is : ',sum1)
sum_of_tuple(l1)
```

```
Sum of first element in tuple is :  15
```

08) Write a recursive function to find nth term of Fibonacci Series.

In [106…
```
n=int(input('Enter n : '))
def fibo(n):
    if n<=1:
        return n
    else:
        return fibo(n-1)+fibo(n-2)
fibo(n)
```

Out[106… 21

09) Write a function to get the name of the student based on the given rollno.

Example: Given dict1 = {101:'Ajay', 102:'Rahul', 103:'Jay', 104:'Pooja'} find name of student whose rollno = 103

In [110…
```
dict1 = {101:'Ajay', 102:'Rahul', 103:'Jay', 104:'Pooja'}
rollNo=int(input('Enter roll no : '))
def get_name(dict1,rollNo):
    return dict1.get(rollNo)
get_name(dict1,rollNo)
```

Out[110… 'Ajay'

10) Write a function to get the sum of the scores ending with zero.

Example : scores = [200, 456, 300, 100, 234, 678]

Ans = 200 + 300 + 100 = 600

In [112…
```
scores = [200, 456, 300, 100, 234, 678]
def sum_of_score(scores):
    total=0
    for i in scores:
        if(i%10==0):
```

```
            total=total+i
    return total
sum_of_score(scores)
```

Out[112… 600

11) Write a function to invert a given Dictionary.

hint: keys to values & values to keys

Before : {'a': 10, 'b':20, 'c':30, 'd':40}

After : {10:'a', 20:'b', 30:'c', 40:'d'}

In [124…
```
dict2={'a': 10, 'b':20, 'c':30, 'd':40}
def invert_dict(dict2):
    dict3={}
    for i in dict2:
        key=i
        val=dict2.get(i)
        dict3[val]=key
    return dict3
invert_dict(dict2)
```

Out[124… {10: 'a', 20: 'b', 30: 'c', 40: 'd'}

12) Write a function to check whether the given string is Pangram or not.

hint: Pangram is a string containing all the characters a-z atlest once.

"the quick brown fox jumps over the lazy dog" is a Pangram string.

In [138…
```
# string=input('Enter string : ')
string='the quick brown fox jumps over the lazy dog" is a Pangram string'
alpha=[]
for i in range(ord('a'),ord('z')+1):
    alpha.append(chr(i))

def check_pangram(string):
        for j in alpha:
            if(j not in string):
                return 'String is not pangram string'
        return 'String is pangram string'
check_pangram(string)
```

Out[138… 'String is pangram string'

13) Write a function that returns the number of uppercase and lowercase letters in the given string.

example : Input : s1 = AbcDEfgh ,Ouptput : no_upper = 3, no_lower = 5

In [146…
```
s2='AbcDEfgh'
print(ord('a'))
```

```
print(ord('A'))
print(ord('z'))
print(ord('Z'))

def count_case(s2):
    no_upper=0
    no_lower=0
    for i in s2:
        if(ord(i)>=97 and ord(i)<=122):
            no_lower+=1
        else:
            no_upper+=1
    print('number of lower case is : ',no_lower)
    print('number of upper case is : ',no_upper)
count_case(s2)
```

```
97
65
122
90
number of lower case is :  5
number of upper case is :  3
```

14) Write a lambda function to get smallest number from the given two numbers.

In [5]:
```
n1=int(input('Enter num 1 :'))
n2=int(input('Enter num 2 :'))
small=lambda n1,n2:n2 if n1>n2 else n1
small(n1,n2)
```

Out[5]:
```
2
```

15) For the given list of names of students, extract the names having more that 7 characters. Use filter().

In [15]:
```
names=['Tanisha','pragati','Virangi','vrushika','pushti']
ans=list(filter(lambda x:len(x) > 7, names))
print(ans)
```

```
['vrushika']
```

16) For the given list of names of students, convert the first letter of all the names into uppercase. use map().

In [27]:
```
names=['tanisha','pragati','virangi','vrushika','pushti']
ans=list(map(lambda x:x.capitalize(),names))
ans
```

Out[27]:
```
['Tanisha', 'Pragati', 'Virangi', 'Vrushika', 'Pushti']
```

17) Write udfs to call the functions with following types of arguments:

1. Positional Arguments

2. Keyword Arguments
3. Default Arguments
4. Variable Legngth Positional(*args) & *variable length Keyword Arguments (*kwargs)*
5. Keyword-Only & Positional Only Arguments

In [55]:
```
def position_agrs(name,age):
    print(f'name is {name} and age {age}')

position_agrs('Nihar',12)

def keyword_agrs(name,age):
    print(f'name is {name} and age {age}')

keyword_agrs(name='Tanisha',age=18)

def default_agrs(name,age=18):
    print(f'name is {name} and age {age}')

default_agrs(name='Alice')
default_agrs(name='Alice',age=20)

def variable_len_positional(*args):
    print(args)

variable_len_positional(1,2,3)

def variable_len_key(**kwards):
    print(kwards)

variable_len_key(name='tanisha',age=12)

def keyword_only(name,*,age):
    print(f'name is {name} and age is {age}')

keyword_only('tanisha',age=25)

def positional_only(name,/,age):
    print(f'name is {name} and age is {age}')

positional_only('tanisha',19)
```

```
name is Nihar and age 12
name is Tanisha and age 18
name is Alice and age 18
name is Alice and age 20
(1, 2, 3)
{'name': 'tanisha', 'age': 12}
name is tanisha and age is 25
name is tanisha and age is 19
```

In [ ]:

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Tanisha Bhalodiya - 23010101021

Lab - 9

File I/O

01) WAP to read and display the contents of a text file. (also try to open the file in some other directory)

- in the form of a string

- line by line

- in the form of a list

In [35]:
```
fp=open('file1.txt','r')
print('...In form of string.................')
data=fp.read()
print(data)
fp.close()
print(type(data))

print('line by line..............')
fp=open('file1.txt','r')
print(fp.readline())
fp.close()

print('in the form of a list...........')
fp=open('file1.txt','r')
li=fp.readlines()
# print(li)

for i in li:
    print(i,end='')
fp.close()
```

```
...In form of string..................
hi.
hello from python...
how are you ?
<class 'str'>
line by line..............
hi.

in the form of a list...........
hi.
hello from python...
how are you ?
```

02) WAP to create file named "new.txt" only if it doesn't exist.

In [41]:
```
fp=open('new.txt','w')
fp.close()
```

03) WAP to read first 5 lines from the text file.

In [43]:
```
fp=open('file1.txt','r')
for i in range(5):
    print(fp.readline())
fp.close()
```

```
hi.

hello from python...

how are you ?

Python is a high-level,

general-purpose programming language.
```

04) WAP to find the longest word(s) in a file

In [49]:
```
with open('file1.txt', 'r') as file:
        words = file.read().split()
max_len=0
for i in words:
    if max_len<len(i) :
        max_len=len(i)
        longest_words=[i]
    elif len(i) == max_len:
        longest_words.append(i)

# max_length = max(len(word) for word in words)
# longest_words = [word for word in words if len(word) == max_length]
print(longest_words)
fp.close()
```

```
['general-purpose']
```

05) WAP to count the no. of lines, words and characters in a given text file.

In [63]:
```
lines=0
words=0
character=0
with open('file1.txt','r') as file:
    for i in file:
        lines+=1

        words_array=i.split()
        words=len(words_array)

        character=len(i)
print('Number of lines : ',lines)
print('Number of words : ',words)
print('Number of characters : ',character)
```

```
Number of lines :  7
Number of words :  7
Number of characters :  52
```

06) WAP to copy the content of a file to the another file.

In [65]:
```
fp1=open('file1.txt','r')
fp2=open('file2.txt','w')
data=fp1.read()
fp2.write(data)
fp1.close()
fp2.close()
```

07) WAP to find the size of the text file.

In [67]:
```
with open('file1.txt','rb') as file:
    # take cursor on the end of file
    file.seek(0,2)
    size_of_file=file.tell()
print('size o file is : ',size_of_file)
```

```
size o file is :  194
```

08) WAP to create an UDF named frequency to count occurances of the specific word in a given text file.

In [85]:
```
def frequency(filename,word):
    c=0
    with open(filename,'r') as file:
        for i in file:
            words_arr=i.split()
            c+=words_arr.count(word)
        print(f'Occurances of word {word} is {c}')
frequency('file1.txt','python')
```

```
Occurances of word python is 1
```

09) WAP to get the score of five subjects from the user, store them in a file. Fetch those marks and find the highest score.

In [121…
```
max_score=0
mark=[]
for i in range(5):
    m=int(input('Enter mark'))
    mark.append(m)

with open('marks.txt','w') as file:
    for i in mark:
        file.write(f'{i}\n')

print('Marks is written')

with open('marks.txt','r') as file:
    mark=[float(i.strip()) for i in file]

print('Highest score is : ',max(mark))
```

```
Marks is written
Highest score is :  50.0
```

10) WAP to write first 100 prime numbers to a file named primenumbers.txt

(Note: each number should be in new line)

In [158…
```
prime = []
for n in range(2, 101):
    for i in range(2, int(n/2) + 1):
        if n % i == 0:
            break
    else:
        prime.append(n)

with open('primenumbers.txt', 'w') as file:
    for p in prime:
        file.write(f"{p}\n")

print("The prime numbers written to 'primenumbers.txt'.")
```

```
The prime numbers written to 'primenumbers.txt'.
```

11) WAP to merge two files and write it in a new file.

In [162…
```
with open('file3.txt','w+') as f1:
    with open('file1.txt','r') as f2:
        f1.write(f2.read())
    with open('file2.txt','r') as f3:
        f1.write(f3.read())
```

12) WAP to replace word1 by word2 of a text file. Write the updated data to new file.

In [180…
```
word1=input('Enter word 1 : ')
word2=input('Enter word 2 : ')
with open('file4.txt','w+') as f1:
    with open('file3.txt','r') as f2:
        data=f2.read()
    updated_data=data.replace(word1,word2)
    f1.write(updated_data)
```

13) Demonstrate tell() and seek() for all the cases(seek from beginning-end-current position) taking a suitable example of your choice.

In [212…
```
with open('file1.txt','r') as file:
    print(file.tell())
    file.seek(5)
    print(file.tell())
```

```
0
5
```

In [5]:
```
# EXCEPTION HANDLING

class Person:
    def __init__(self,name,age):
        self.name=name
        self.age=age


p1=Person('xyz',20)
print(p1.name)
print(p1.age)
```

```
xyz
20
```

In [ ]:
```

```

In [21]:
```
print('before try block')
try:
    a=int(input('Enter n : '))
    print(a/1)
except ZeroDivisionError:
    print('Some error has generate')
except:
    print('DEFAULT ERROR')
print('terminate')
```

```
before try block
5.0
terminate
```

In [ ]:
```

```

Darshan UNIVERSITY
योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Tanisha Bhalodiya - 23010101021

Lab - 10

Exception Handling

01) WAP to handle following exceptions: 1. ZeroDivisionError 2. ValueError 3. TypeError #### Note: handle them using separate except blocks and also using single except block too.

In [15]:
```
# USING SEPARATE EXCEPT BLOCK
try:
    n1=int(input('Enter number 1 : '))
    n2=int(input('Enter number 2 : '))
    ans=n1/n2
    print('answer is : ',ans)
except ZeroDivisionError:
    print('zero division error')
except ValueError:
    print('Value error occur')
except TypeError:
    print('Type error')
```

```
Value error occur
```

In [7]:
```
# USING SINGLE EXCEPT BLOCK
try:
    a=int(input('Enter number a : '))
    b=int(input('Enter number b : '))
    print(a/b)

except (ZeroDivisionError,ValueError,TypeError) as err:
    print(f'Error  : {err}')
```

```
Error  : division by zero
```

02) WAP to handle following exceptions:

1. IndexError
2. KeyError

In [25]:
```
# INDEX ERROR
try:
    l1=[1,2,3,4,5]
    i=int(input('Enter index : '))
    print(l1[i])
except IndexError:
    print('Index error occur')

# KEY ERROR
try:
    m1={'name':'Kartik','age':25}
    key=input('Enter key : ')
    print(m1[key])
except KeyError:
    print('Key error occur')c
```

```
Index error occur
Key error occur
```

03) WAP to handle following exceptions:

1. FileNotFoundError
2. ModuleNotFoundError

In [3]:
```
# FILE NOT FOUND ERROR
try:
    fp=open('xyz.txt','r')
except FileNotFoundError:
    print('File not found please make it first')


# Module not found error
try:
    import maths
except ModuleNotFoundError:
    print('Module not found please check it')
```

```
File not found please make it first
Module not found please check it
```

04) WAP that catches all type of exceptions in a single except block.

In [35]:
```
try:
    a=int(input('Enter number 1 : '))
    b=int(input('Enter number 2 : '))
    sum1=a+b
    div=a/b
    print('division is : ',div)
    print('Sum is : ',sum1)
except Exception as e:
    print('error occured : ',e)
```

```
error occured :  division by zero
```

05) WAP to demonstrate else and finally block.

In [50]:
```
try:
    a=int(input('Enter number 1 : '))
    b=int(input('Enter number 2 : '))
    div=a/b
except ZeroDivisionError as err:
    print(err)
else:
    print('division is : ',div)
finally:
    print('Finally block which always executed')
```

```
division by zero
Finally block which always executed
```

06) Create a short program that prompts the user for a list of grades separated by commas.

Split the string into individual grades and use a list comprehension to convert each string to an integer.

You should use a try statement to inform the user when the values they entered cannot be converted.

In [60]:
```
try:
    str1=input('Enter string of grade seperated by comma : ')
    grades=[int(g) for g in str1.split(',')]
    print('We have grades : ',grades)
except Exception as e:
    print('Error is : ',e)
```

```
Error is :  invalid literal for int() with base 10: 'dhbh'
```

07) WAP to create an udf divide(a,b) that handles ZeroDivisionError.

In [64]:
```
def divide(a,b):
    try:
        res=a/b
        return res
    except ZeroDivisionError:
        return 'division by zero is not allowed'

a=int(input('Enter a : '))
b=int(input('Enter b : '))
ans=divide(a,b)
print(ans)
```

```
division by zero is not allowed
```

08) WAP that gets an age of a person form the user and raises ValueError with error message: "Enter Valid Age" :

If the age is less than 18.

otherwise print the age.

In [76]:
```
try:
    age=int(input('Enter age : '))
    if age>=18:
        print('Your age is : ',age)
    else:
        raise ValueError('Enter valid age')
except ValueError as err:
    print(err)
```

```
Enter valid age
```

09) WAP to raise your custom Exception named InvalidUsernameError with the error message : "Username must be between 5 and 15 characters long":

if the given name is having characters less than 5 or greater than 15.

otherwise print the given username.

In [96]:
```
name=input('Enter name : ')
try:
    length=len(name)
    if length<5 or length>15:
        raise InvalidUsernameError('Username must be between 5 and 15 characters
    else:
        print('Your name is : ',name)
except InvalidUsernameError as err:
    print(err)
```

```
Username must be between 5 and 15 characters long
```

10) WAP to raise your custom Exception named NegativeNumberError with the error message : "Cannot calculate the square root of a negative number" : if the given number is negative. otherwise print the square root of the given number.

In [118…
```
num=int(input('Enter number : '))
import math
try:
    if num>0:
        print('square root of number is : ',math.sqrt(num))
    else:
        raise NegativeNumberError('Cannot calculate the square root of a negativ
except NegativeNumberError as err:
    print(err)
```

```
Cannot calculate the square root of a negative number
```

In [ ]:
```
```

Python Programming - 2301CS404

Tanisha Bhalodiya - 23010101021

Lab - 11

Modules

01) WAP to create Calculator module which defines functions like add, sub,mul and div.

Create another .py file that uses the functions available in Calculator module.

In [23]:
```
import calculator


n1=int(input('Enter number 1 : '))
n2=int(input('Enter number 2 : '))

c=int(input('Enter 1 for addition , 2 for substraction , 3 for multiplication an

if c==1:
    print(f"Result of sum is : {calculator.add(n1, n2)}")
elif c==2:
    print(f"Result of substration is : {calculator.sub(n1, n2)}")
elif c==3:
    print(f"Result of multiplication is : {calculator.mul(n1, n2)}")
elif c==4:
    print(f"Result of division is : {calculator.div(n1, n2)}")
else:
    print('INVALID CHOICE')
```

```
Result of multiplication is : 10
```

02) WAP to pick a random character from a given String.

In [27]:
```
import random
str1=input('Enter string : ')
```

```
if len(str1)==0:
    print('String is empty')
else:
    print(f'Random variable is : {random.choice(str1)}')
```

```
Random variable is : a
```

03) WAP to pick a random element from a given list.

In [37]:
```
import random
l1=[10,20,30,40,50]
print(f'Random element from list is : {random.choice(l1)}')
```

```
Random element from list is : 40
```

04) WAP to roll a dice in such a way that every time you get the same number.

In [53]:
```
import random
random.seed(4)
print(f'roll a dice in such a way that every time you get the same number : {ran
```

```
roll a dice in such a way that every time you get the same number : 2
```

05) WAP to generate 3 random integers between 100 and 999 which is divisible by 5.

In [63]:
```
import random
l1=[]
while len(l1)<3:
    a=random.randint(100,999)
    if a%5==0:
        l1.append(a)
print('3 random integers : ',l1)
```

```
3 random integers :  [660, 960, 620]
```

06) WAP to generate 100 random lottery tickets and pick two lucky tickets from it and announce them as Winner and Runner up respectively.

In [79]:
```
import random
tick=[]
while len(tick)<100:
    t=random.randint(1000,9999)
    tick.append(t)

lucky=random.sample(tick,2)
winner=lucky[0]
runner_up=lucky[1]

print(f'Winner is {winner} and Runner_up is {runner_up}')
```

```
Winner is 7030 and Runner_up is 9030
```

07) WAP to print current date and time in Python.

In [89]:
```
import datetime
d1=datetime.datetime.now()
print('Current date and time is : ',d1)
```

```
Current date and time is :  2025-02-12 18:30:58.348167
```

08) Subtract a week (7 days) from a given date in Python.

In [93]:
```
from datetime import datetime,timedelta
today=datetime.today()
last_week=today-timedelta(7)

print('today the date is : ',today.date())
print('a week before date is : ',last_week.date())
```

```
today the date is :  2025-02-12
a week before date is :  2025-02-05
```

09) WAP to Calculate number of days between two given dates.

In [97]:
```
from datetime import datetime
d1=input('Enter date 1 : ')
d2=input('Enter date 2 : ')

# convert string into date object
diff1=datetime.strptime(d1,'%Y-%m-%d')
diff2=datetime.strptime(d2,'%Y-%m-%d')
day=abs((diff1-diff2).days)

print('Number of days between two given dates : ',day)
```

```
Number of days between two given dates :  31
```

10) WAP to Find the day of the week of a given date.(i.e. wether it is sunday/monday/tuesday/etc.)

In [105…
```
from datetime import datetime
d1=input('Enter date : ')
date=datetime.strptime(d1,'%Y-%m-%d')

day=date.strftime('%A')

print('day of week in given date : ',day)
```

```
day of week in given date :  Wednesday
```

11) WAP to demonstrate the use of date time module.

In [107…
```
from datetime import datetime, timedelta

# Get current date and time
now = datetime.now()
print("Current Date and Time:", now)

# Get today's date
today = datetime.today().date()
```

```
print("Today's Date:", today)

# Format the date
formatted_date = now.strftime("%d-%m-%Y %H:%M:%S")
print("Formatted Date and Time:", formatted_date)

# Get day of the week
day_of_week = now.strftime("%A")
print("Day of the Week:", day_of_week)

# Add 5 days to the current date
future_date = now + timedelta(days=5)
print("Date After 5 Days:", future_date.date())

# Subtract 7 days from the current date
past_date = now - timedelta(days=7)
print("Date 7 Days Ago:", past_date.date())

# Calculate difference between two dates
date1 = datetime(2025, 1, 1)
date2 = datetime(2025, 2, 12)
difference = date2 - date1
print("Difference Between Dates:", difference.days, "days")
```

```
Current Date and Time: 2025-02-12 18:54:52.648309
Today's Date: 2025-02-12
Formatted Date and Time: 12-02-2025 18:54:52
Day of the Week: Wednesday
Date After 5 Days: 2025-02-17
Date 7 Days Ago: 2025-02-05
Difference Between Dates: 42 days
```

12) WAP to demonstrate the use of the math module.

In [109…

```
import math

# Find the Square Root
num = 25
sqrt = math.sqrt(num)
print(f"Square root of {num} is: {sqrt}")

# Find the Power
base = 2
exp = 3
power = math.pow(base, exp)
print(f"{base} raised to the power of {exp} is: {power}")

# Find the Greatest Common Divisor (GCD)
a = 60
b = 48
gcd = math.gcd(a, b)
print(f"GCD of {a} and {b} is: {gcd}")

# Find the Factorial
n = 5
fact = math.factorial(n)
print(f"Factorial of {n} is: {fact}")

# Find the Value of Pi and Euler's Number
```

```
print("Value of Pi:", math.pi)
print("Value of Euler's Number (e):", math.e)

# Use Trigonometric Functions
angle = 90
radians = math.radians(angle)  # Convert degrees to radians
sin_val = math.sin(radians)
cos_val = math.cos(radians)
print(f"Sine of {angle} degrees is: {sin_val}")
print(f"Cosine of {angle} degrees is: {cos_val}")

# Rounding Functions
num = 3.75
print(f"Floor of {num} is: {math.floor(num)}")
print(f"Ceiling of {num} is: {math.ceil(num)}")
```

```
Square root of 25 is: 5.0
2 raised to the power of 3 is: 8.0
GCD of 60 and 48 is: 12
Factorial of 5 is: 120
Value of Pi: 3.141592653589793
Value of Euler's Number (e): 2.718281828459045
Sine of 90 degrees is: 1.0
Cosine of 90 degrees is: 6.123233995736766e-17
Floor of 3.75 is: 3
Ceiling of 3.75 is: 4
```

In [ ]:

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

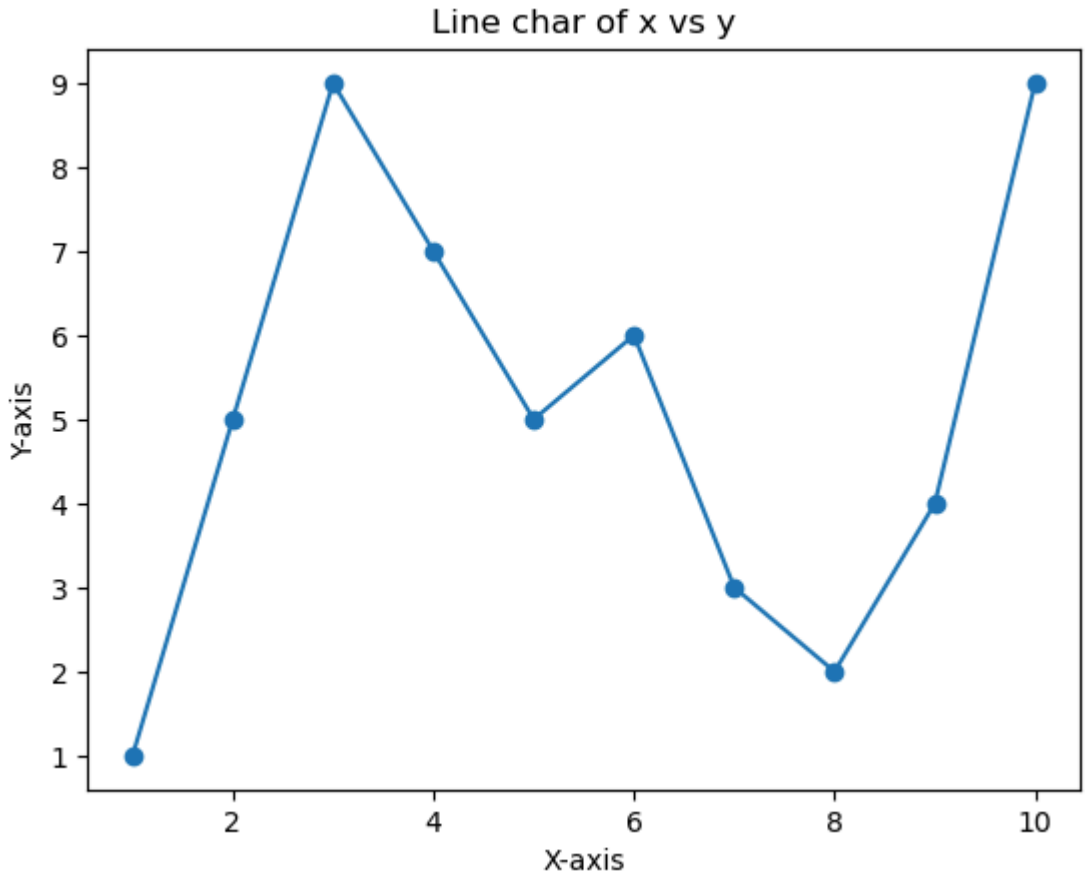
Python Programming - 2301CS404

Tanisha Bhalodiya - 23010101021

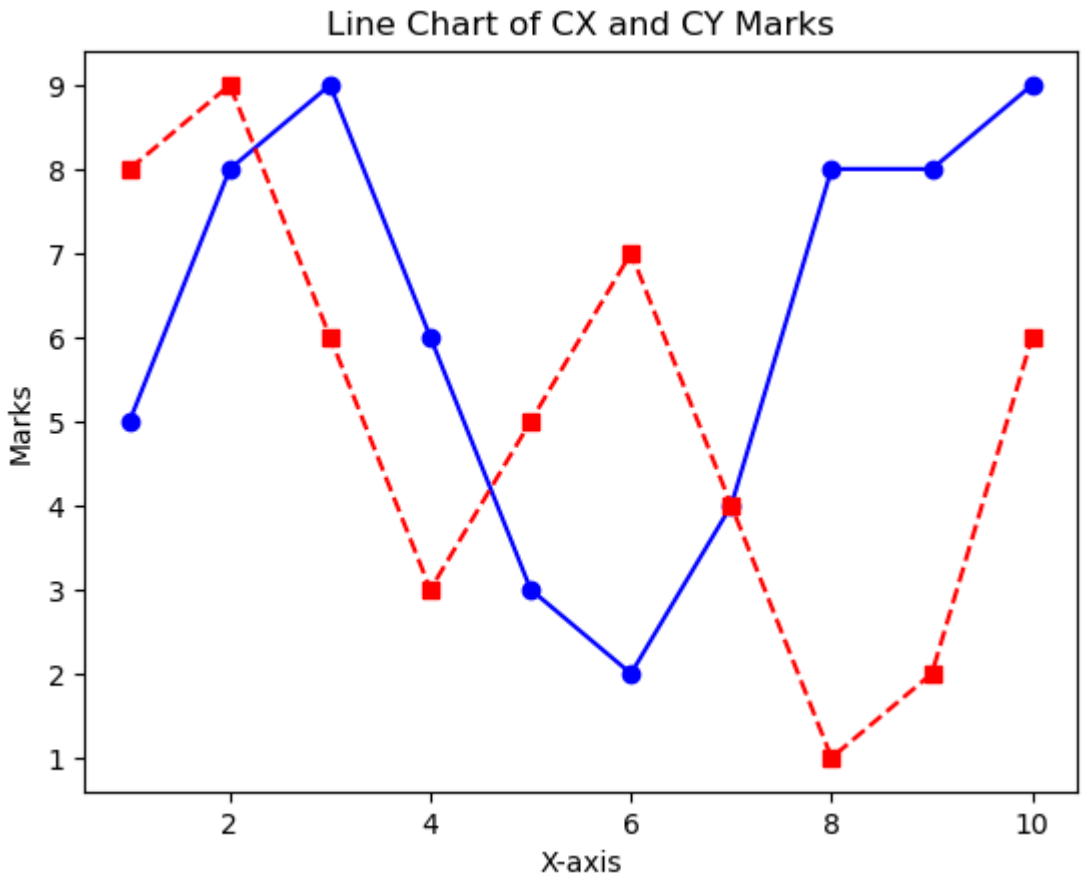
Lab - 12

In [1]:
```
#import matplotlib below
import matplotlib.pyplot as plt
```

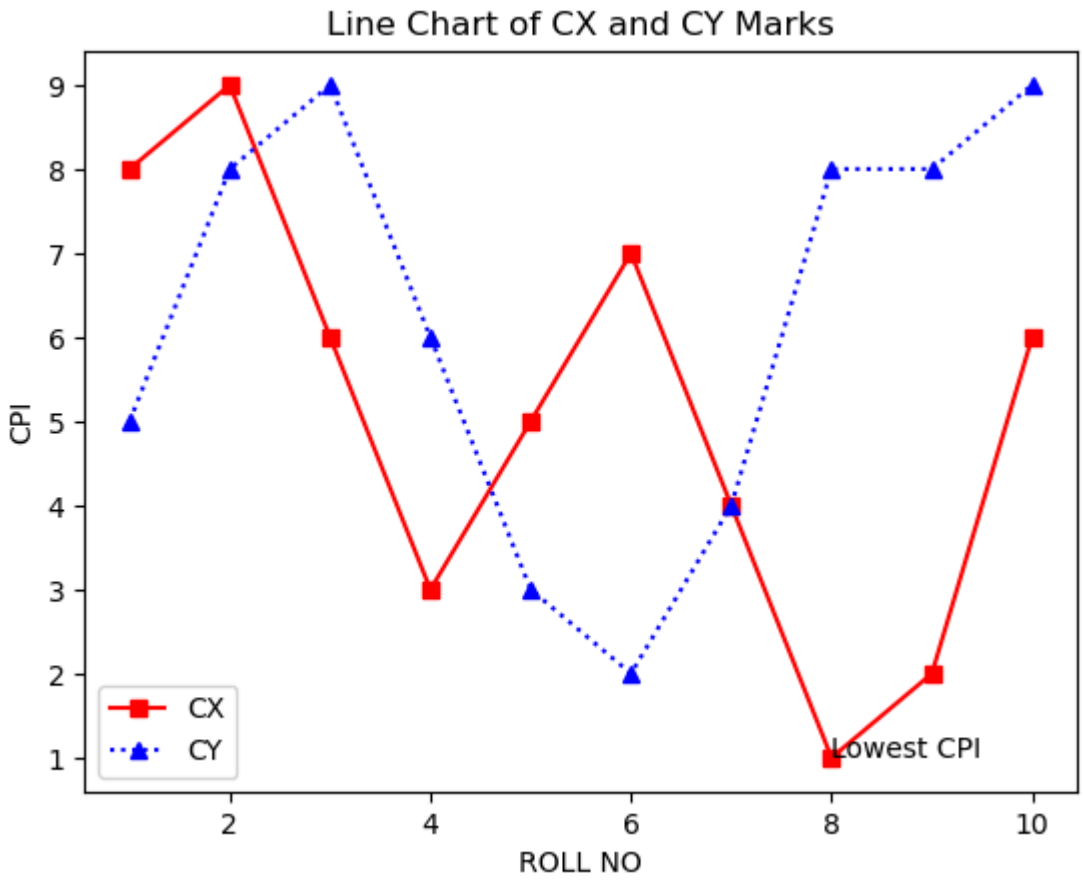
In [14]:
```
x = range(1,11)
y = [1,5,9,7,5,6,3,2,4,9]

# write a code to display the line chart of above x & y
plt.plot(x,y,marker='o',linestyle='-')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.title('Line char of x vs y')
plt.show()
```



In [17]:
```
x = [1,2,3,4,5,6,7,8,9,10]
cxMarks = [5,8,9,6,3,2,4,8,8,9]
cyMarks = [8,9,6,3,5,7,4,1,2,6]

# write a code to display two lines in a line chart (data given above)
plt.plot(x,cxMarks,marker='o',linestyle='-',color='b',label='XC line')
plt.plot(x,cyMarks,marker='s',linestyle='--',color='r',label='YC line')
plt.xlabel('X-axis')
plt.ylabel('Marks')
plt.title('Line Chart of CX and CY Marks')
plt.show()
```



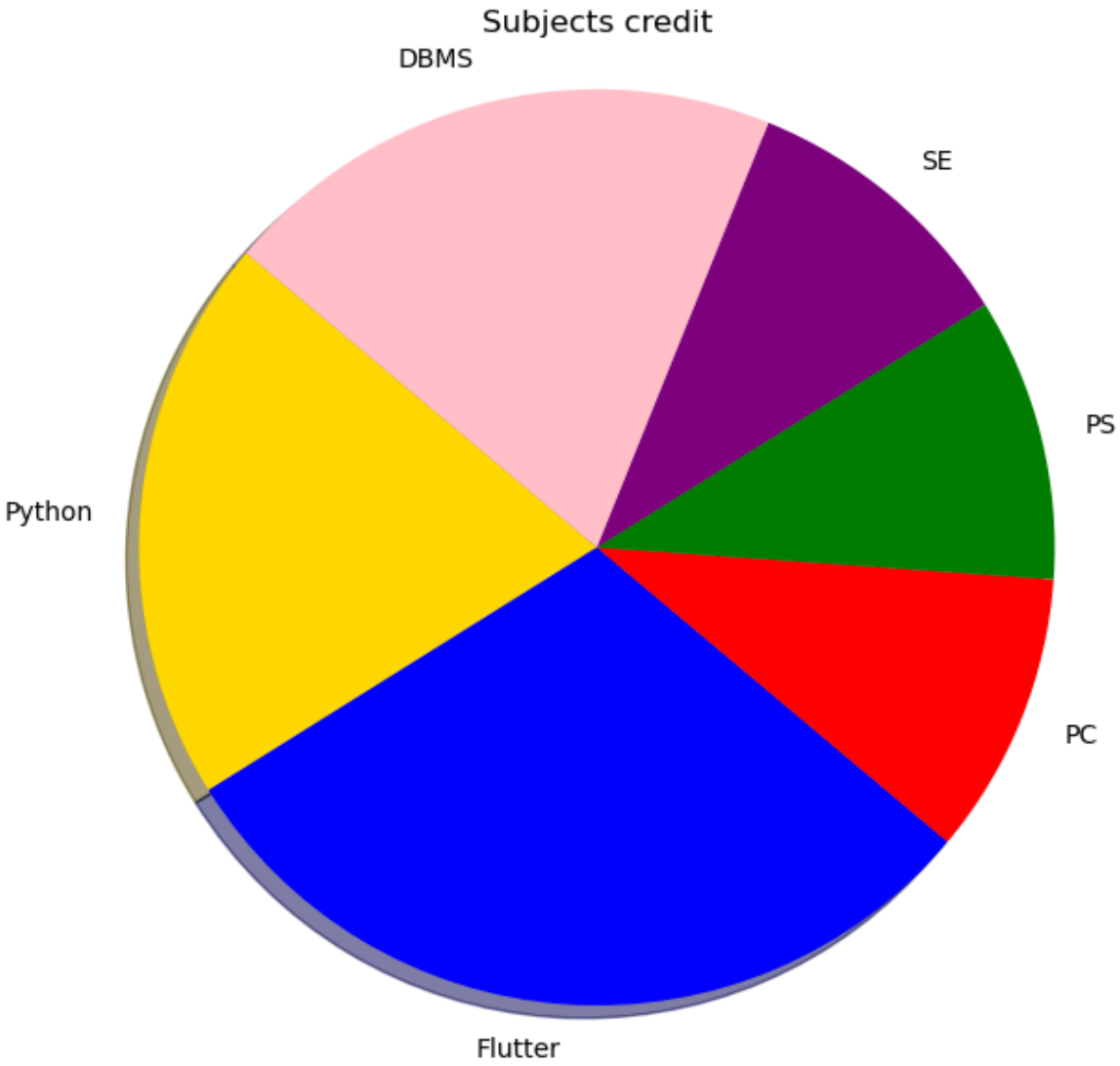
In [7]:
```
x = range(1,11,1)
cxMarks= [8,9,6,3,5,7,4,1,2,6]
cyMarks= [5,8,9,6,3,2,4,8,8,9]


# write a code to generate below graph
plt.plot(x,cxMarks,linewidth=1.5,marker='s',linestyle='-',color='r',label='CX')
plt.plot(x,cyMarks,linewidth=1.5,marker='^',linestyle=':',color='b',label='CY')
plt.xlabel('ROLL NO')
plt.ylabel('CPI')
plt.title('Line Chart of CX and CY Marks')
plt.text(8, 1, "Lowest CPI", fontsize=10, color='black')

plt.legend()
plt.show()
```



In [ ]:

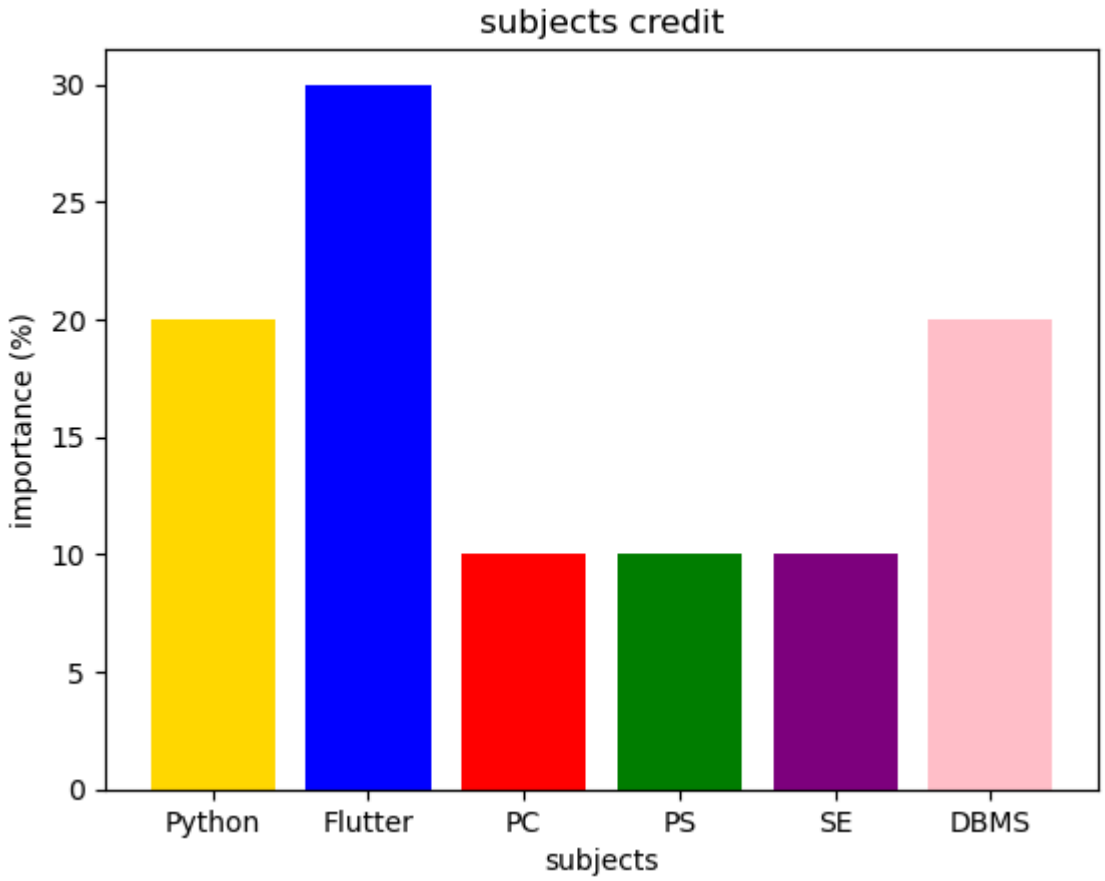
04) WAP to demonstrate the use of Pie chart.

In [20]:
```
labels = ['Python', 'Flutter', 'PC', 'PS', 'SE','DBMS']
size=[20,30,10,10,10,20]
credit=[5,6,2,4,5,5]
colors=['gold', 'blue', 'red', 'green', 'purple','pink']

plt.figure(figsize=(7,7))
plt.pie(size,labels=labels,colors=colors, startangle=140, shadow=True)
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.title("Subjects credit")
plt.show()
```



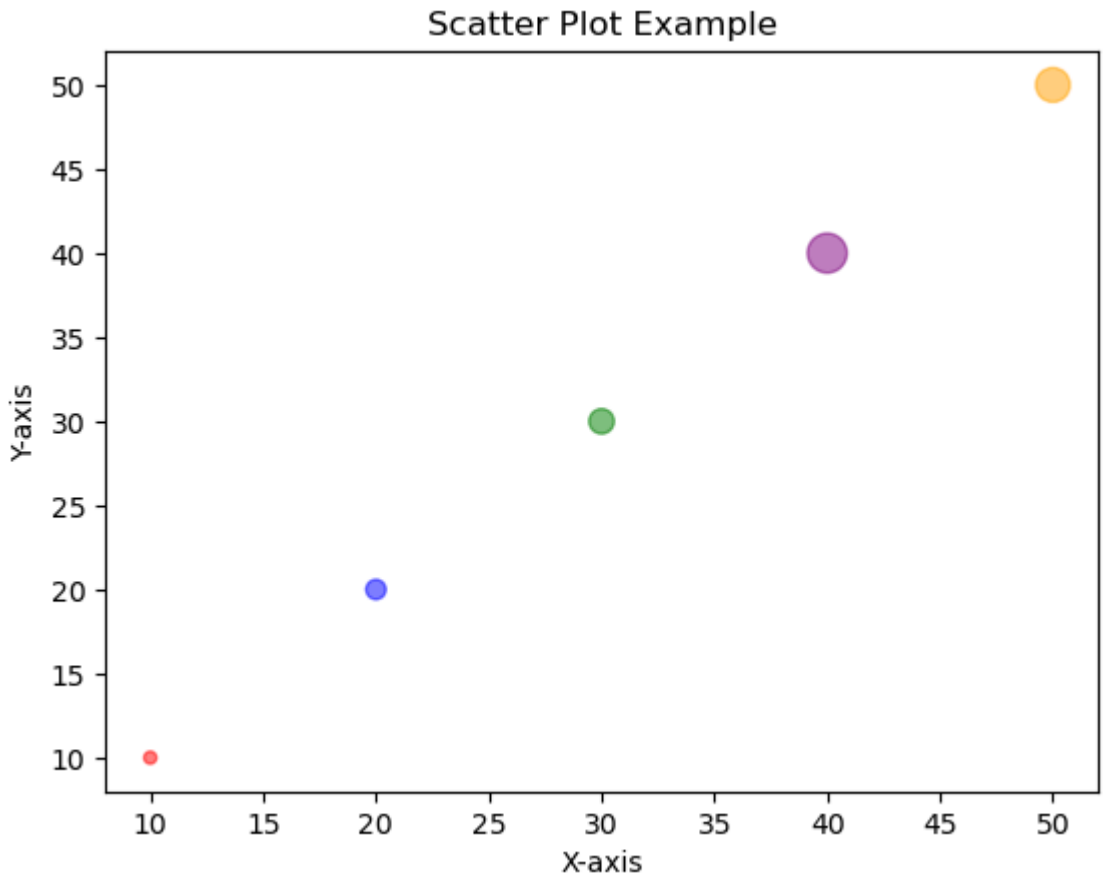
05) WAP to demonstrate the use of Bar chart.

In [24]:
```
labels = ['Python', 'Flutter', 'PC', 'PS', 'SE','DBMS']
size=[20,30,10,10,10,20]
colors=['gold', 'blue', 'red', 'green', 'purple','pink']

plt.bar(labels,size,color=colors)
plt.xlabel("subjects")
plt.ylabel("importance (%)")
plt.title("subjects credit")
plt.show()
```



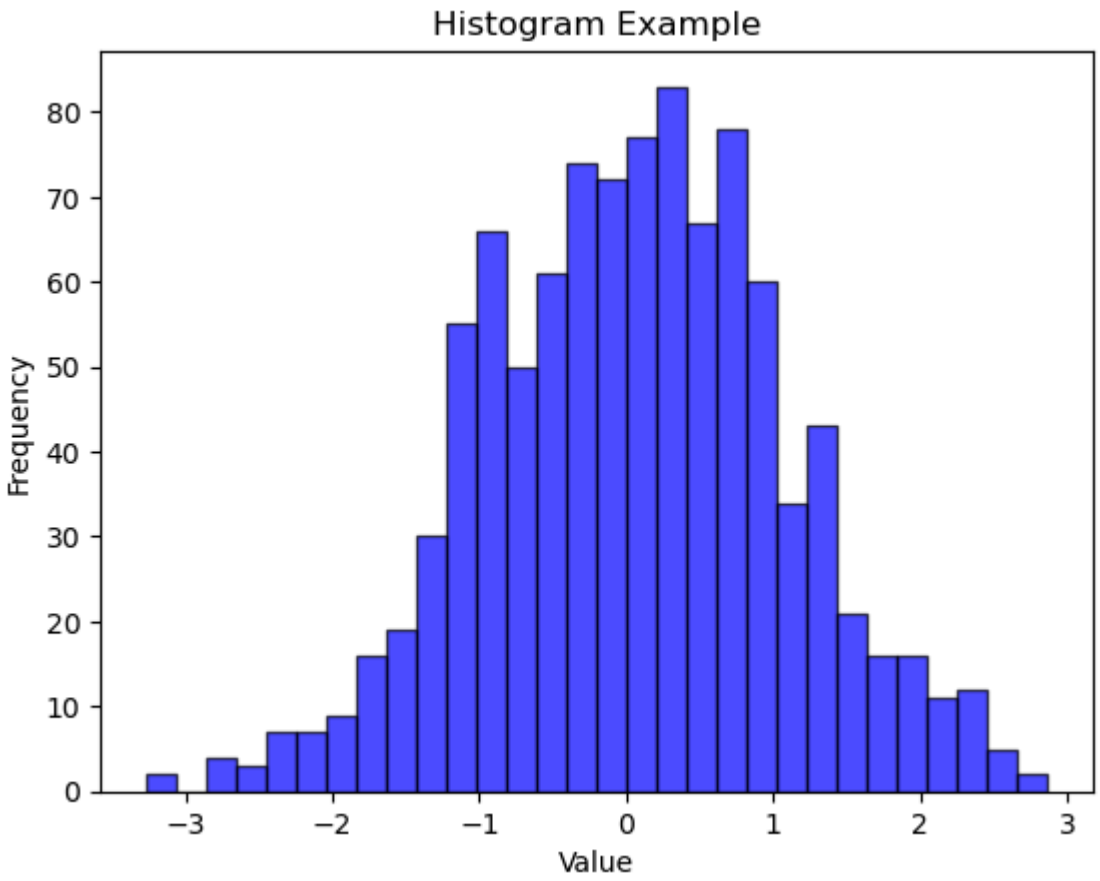
06) WAP to demonstrate the use of Scatter Plot.

In [28]:
```
# Data to plot
x = [10, 20, 30, 40, 50]
y = [10, 20, 30, 40, 50]
colors = ['red', 'blue', 'green', 'purple', 'orange']
sizes = [20, 50, 80, 200, 150]

# Plot scatter plot
plt.scatter(x, y, c=colors, s=sizes, alpha=0.5)

plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Scatter Plot Example")
plt.show()
```



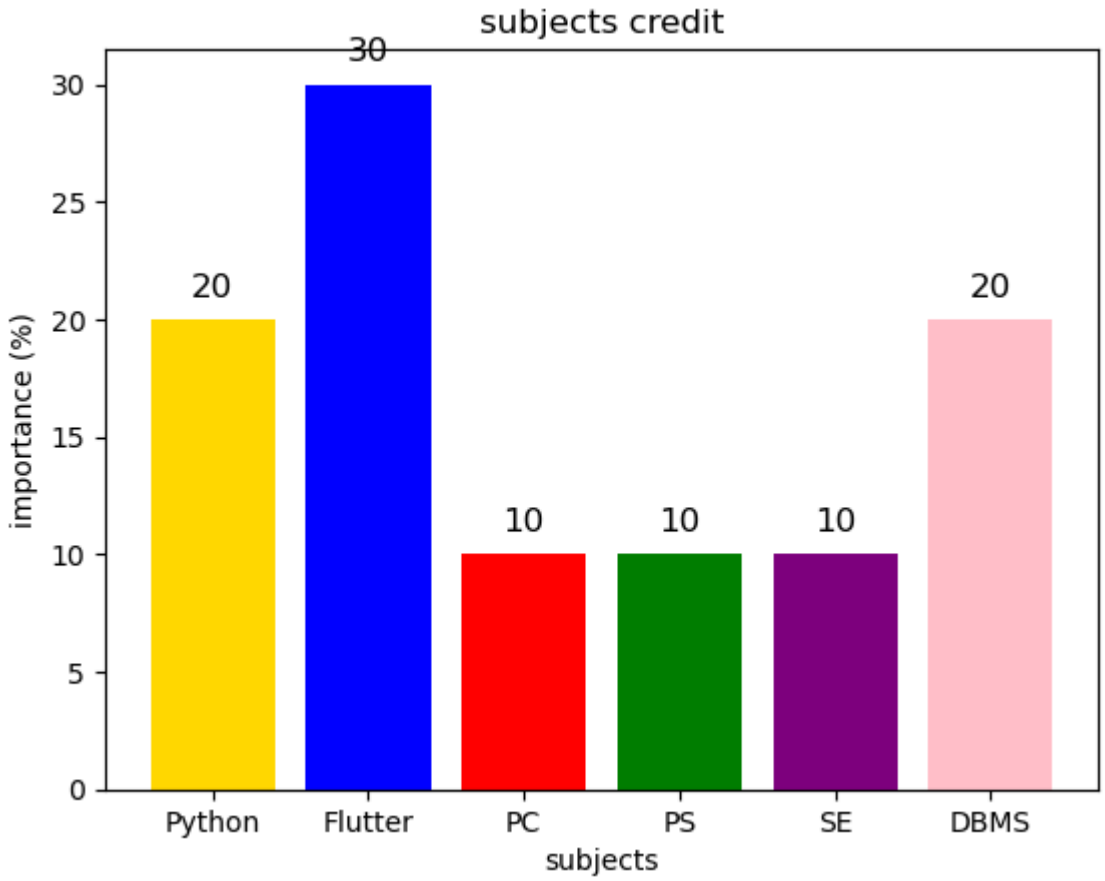
07) WAP to demonstrate the use of Histogram.

In [32]:
```
import numpy as np
data = np.random.randn(1000)  # Generate random number data

# Plot histogram
plt.hist(data, bins=30, color='blue', edgecolor='black', alpha=0.7)

plt.xlabel("Value")
plt.ylabel("Frequency")
plt.title("Histogram Example")
plt.show()
```



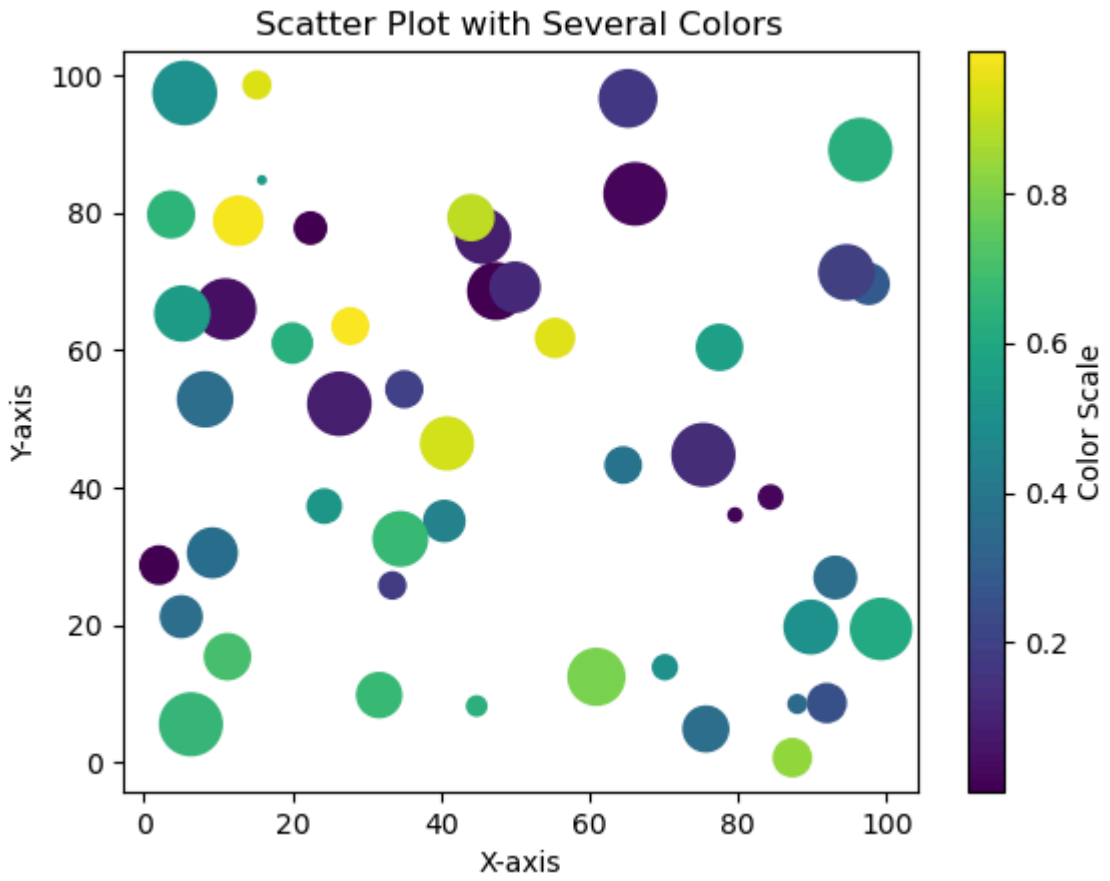
08) WAP to display the value of each bar in a bar chart using Matplotlib.

In [50]:
```
labels = ['Python', 'Flutter', 'PC', 'PS', 'SE','DBMS']
size=[20,30,10,10,10,20]
colors=['gold', 'blue', 'red', 'green', 'purple','pink']

plt.bar(labels, size, color=colors)

for i , x in enumerate(size):
    plt.text(i,x+1,str(x),ha='center',fontsize=12)

plt.xlabel("subjects")
plt.ylabel("importance (%)")
plt.title("subjects credit")
plt.show()
```



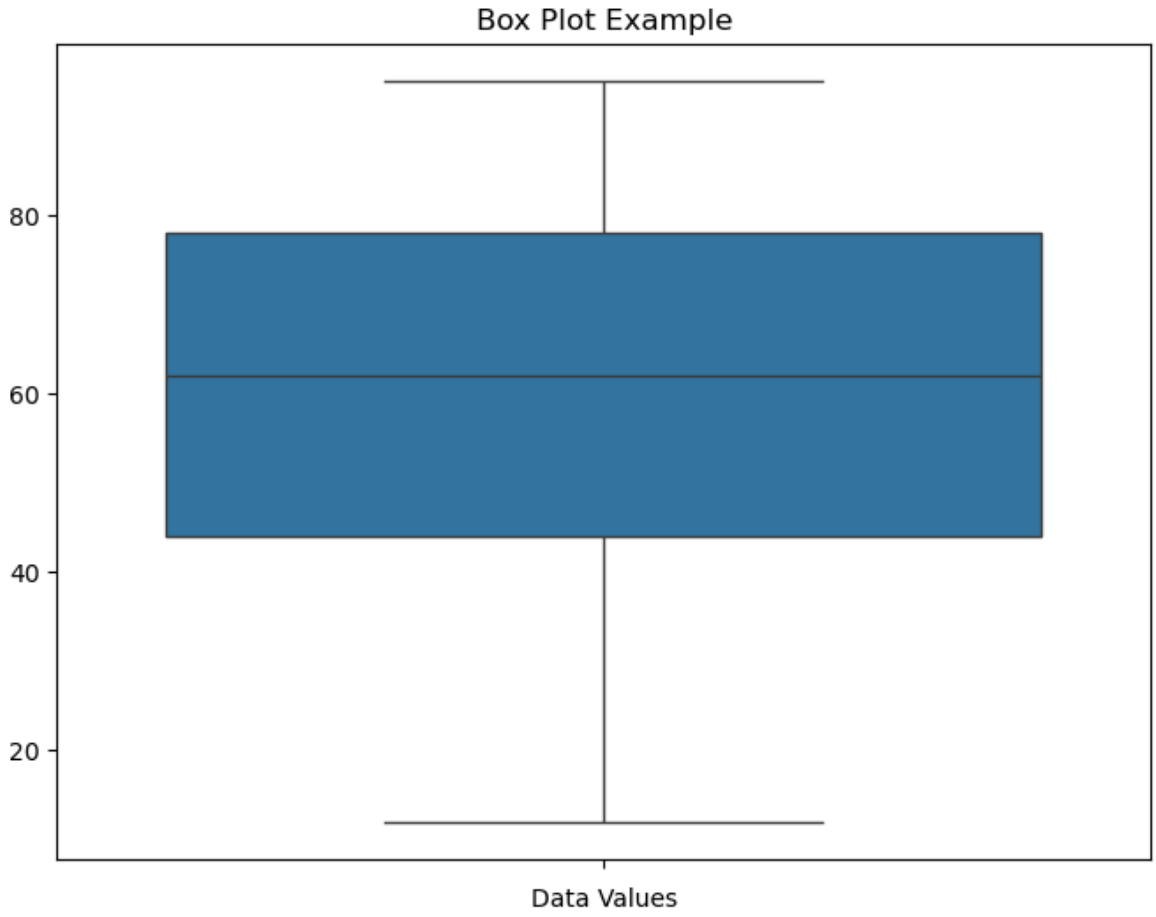
09) WAP create a Scatter Plot with several colors in Matplotlib?

In [58]:
```
x = np.random.rand(50) * 100
y = np.random.rand(50) * 100
colors = np.random.rand(50)
sizes = np.random.rand(50) * 500


plt.scatter(x, y, c=colors, s=sizes)
# , alpha=0.7, cmap='viridis'
plt.colorbar(label="Color Scale")

plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Scatter Plot with Several Colors")
plt.show()
```



10) WAP to create a Box Plot.

In [4]:
```
import seaborn as sns
graph=[12,45,78,45,95,84,62,12,75,35,65,44,87]
plt.figure(figsize=(8,6))
sns.boxplot(data=graph)
plt.title('Box Plot Example')
plt.xlabel('Data Values')
plt.show()
```



In [ ]:

Darshan UNIVERSITY
योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Tanisha Bhalodiya - 23010101021

Lab - 13

OOP

01) Write a Program to create a class by name Students, and initialize attributes like name, age, and grade while creating an object.

In [5]:
```
class Students:
    def __init__(self, name, age, grade):
        self.name = name
        self.age = age
        self.grade = grade
    def info(self):
        print(f'Students name is : {self.name}')
        print(f'Students age is : {self.age}')
        print(f'Students grade is : {self.grade}')

stu1=Students('Tanisha',18,'A')
stu1.info()
```

```
Students name is : Tanisha
Students age is : 18
Students grade is : A
```

02) Create a class named Bank_Account with Account_No, User_Name, Email,Account_Type and Account_Balance data members. Also create a method GetAccountDetails() and DisplayAccountDetails(). Create main method to demonstrate the Bank_Account class.

In [9]:
```
class Bank_Account:
    def __init__(self,account_no,user_name,email,account_type,account_balance):
        self.account_no = account_no
        self.user_name = user_name
```

```
        self.email = email
        self.account_type = account_type
        self.account_balance = account_balance

    def GetAccountDetails(self):
        self.account_no = input("Enter Account Number: ")
        self.user_name = input("Enter User Name: ")
        self.email = input("Enter Email: ")
        self.account_type = input("Enter Account Type (Savings/Current): ")
        self.account_balance = float(input("Enter Account Balance: "))

    def DisplayAccountDetails(self):
        print(f'account number is : {self.account_no}')
        print(f'User name is : {self.user_name}')
        print(f'Email is : {self.email}')
        print(f'Account typpe is : {self.account_type}')
        print(f'Account balance : {self.account_balance}')

def main():
    account1 = Bank_Account("123456789", "Tanisha Patel", "tanipatel@gmail.com",
    account1.DisplayAccountDetails()


main()
```

```
account number is : 123456789
User name is : Tanisha Patel
Email is : tanipatel@gmail.com
Account typpe is : Savings
Account balance : 5000000.0
```

03) WAP to create Circle class with area and perimeter function to find area and perimeter of circle.

In [35]:
```
import math
class Circle:
    def __init__(self,r):
        self.r=r

    def CountArea(self):
        area=math.pi * self.r * self.r
        print(f'Area of circle is : {area}')

    def CountPerimeter(self):
        per=2*math.pi * self.r
        print(f'Perimeter of circle is : {per}')

c=Circle(5)
c.CountArea()
c.CountPerimeter()
```

```
Area of circle is : 78.53981633974483
Perimeter of circle is : 31.41592653589793
```

04) Create a class for employees that includes attributes such as name, age, salary, and methods to update and display employee information.

In [47]:
```
class Employees:
    def __init__(self,name,age,salary):
        self.name=name
        self.age=age
        self.salary=salary

    def displayInformation(self):
        print(f'Employee information is name {self.name} , age is {self.age} and

    def updateInformation(self):
        n=input('Enter name : ')
        a=input('Enter age : ')
        s=input('Enter Salary : ')
        self.name=n
        self.age=a
        self.salary=s

        print(f'Updated data is of emplyee is name {self.name} , age is {self.ag

emp=Employees('tanisha',18,5000000)
emp.displayInformation()
emp.updateInformation()
```

```
Employee information is name tanisha , age is 18 and salary is 5000000
Updated data is of emplyee is name niihar , age is 15 and salary is 2669
```

05) Create a bank account class with methods to deposit, withdraw, and check balance.

In [1]:
```
class BankAccount:
    def __init__(self,account,balance):
        self.account=account
        self.balance=balance

    def deposit(self , amount):
        if amount > 0:
            self.balance += amount
            print(f'Balance is : {self.balance}')
        else:
            print('enter posotive amount')

    def withdraw(self,amount):
        if amount <= self.balance:
            self.balance -= amount
            print(f'Balance is : {self.balance}')
            print(f'withdrawal amount is : {amount}')
        else:
            print('You can not get amount more than you already have')

    def checkBalance(self):
        print(f'Account balance is : {self.balance}')

acc = BankAccount("Tanisha", 10000.0)
acc.deposit(50)
acc.withdraw(30)
acc.checkBalance()
```

```
Balance is : 10050.0
Balance is : 10020.0
withdrawal amount is : 30
Account balance is : 10020.0
```

06) Create a class for managing inventory that includes attributes such as item name, price, quantity, and methods to add, remove, and update items.

In [ ]:
```
class Inventory:
    def __init__(self):
        self.inv={}

    def addInv(self,name,price,quantity):
        self.inv[name]={}
        self.inv[name]={'name':name,'price':price,"quantity":quantity}

    def remove(self,name):
        if name in self.inv.keys():
            self.inv.pop(name)
        else:
            print("Invalid Name :")

inve=Inverntory()
inve.addInv()
```

07) Create a Class with instance attributes of your choice.

In [3]:
```
class Car:
    cars = []

    def __init__(self, name, color, price):
        self.name = name
        self.color = color
        self.price = price

    def addCar(self):
        Car.cars.append(self)
        print(f'New car added: {self.name}, Color: {self.color}, Price: {self.pr
        return self

    def displayCars(self):
        print("Cars in inventory:")
        for car in Car.cars:
            print(f'{car.name}, Color: {car.color}, Price: {car.price}')

c = Car('TATA Nexon', 'black', 1000000)
c.addCar()
c.displayCars()
```

```
New car added: TATA Nexon, Color: black, Price: 1000000
Cars in inventory:
TATA Nexon, Color: black, Price: 1000000
```

08) Create one class student_kit

Within the student_kit class create one class attribute principal name (Mr ABC)

Create one attendance method and take input as number of days.

While creating student take input their name .

Create one certificate for each student by taking input of number of days present in class.

In [10]:
```
class StudentKit:
    principal_name='Mr ABC'

    def __init__(self, student_name):
        self.student_name = student_name
        self.attendance_days = 0

    def attendance(self, days):
        self.attendance_days = days

    def generate_certificate(self):
        print("Certificate of Attendance::::::::::::::::::::")
        print(f"This is to certify that {self.student_name} has attended {self.a
        print(f"Principal: {StudentKit.principal_name}")

stu= StudentKit("Tanisha")
stu.attendance(75)
stu.generate_certificate()
```

```
Certificate of Attendance::::::::::::::::::::
This is to certify that Tanisha has attended 75 days of class.
Principal: Mr ABC
```

09) Define Time class with hour and minute as data member. Also define addition method to add two time objects.

In [16]:
```
class Time:
    def __init__(self,hour,minute):
        self.hour=hour
        self.minute=minute

    def addTime(self, other):
        #count minute first
        mins = self.minute + other.minute
        #count hours
        hours = mins // 60
        new_mins = mins % 60
        new_hours = self.hour + other.hour + hours
        return Time(new_hours, mins)

    def display_time(self):
        print(f"{self.hour} : {self.minute}")

t1=Time(2,30)
t2=Time(3,50)
```

```
res=t1.addTime(t2)
res.display_time()
```

```
6 : 80
```

In [ ]:

Darshan UNIVERSITY
योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Tanisha Bhalodiya - 23010101021

Lab - 13

Continued..

10) Calculate area of a ractangle using object as an argument to a method.

In [1]:
```
class Rectangle:
    def __init__(self,l,w):
        self.l=l
        self.w=w

    def area(self):
        print(f'Area of rectangle is : {self.l*self.w}')

r=Rectangle(5,10)
r.area()
```

```
Area of rectangle is : 50
```

11) Calculate the area of a square.

Include a Constructor, a method to calculate area named area() and a method named output() that prints the output and is invoked by area().

In [5]:
```
class Square:
    def __init__(self,l):
        self.l=l

    def area(self):
        ar=self.l**2
        self.output(ar)

    def output(self,ar):
```

```
        print(f'Area of the square : {ar}')


s=Square(5)
s.area()
```

```
Area of the square : 25
```

12) Calculate the area of a rectangle.

Include a Constructor, a method to calculate area named area() and a method named output() that prints the output and is invoked by area().

Also define a class method that compares the two sides of reactangle. An object is instantiated only if the two sides are different; otherwise a message should be displayed : THIS IS SQUARE.

In [17]:
```
class Rectangle:
    def __init__(self,l,w):
        if self.com(l,w):
            self.l=l
            self.w=w
        else:
            print('THIS IS SQUARE')
            self.l=None
            self.w=None

    def area(self):
        ar=self.l*self.w
        self.output(ar)

    def output(self,ar):
        print(f'Area of the rectangle : {ar}')

    def com(self,l,w):
        if l==w:
            return False
        else:
            return True


rec1=Rectangle(5,10)
rec1.area()

rec2=Rectangle(5,5)
```

```
Area of the rectangle : 50
THIS IS SQUARE
```

13) Define a class Square having a private attribute "side".

Implement get_side and set_side methods to accees the private attribute from outside of the class.

In [21]:
```
class Square:
    def __init__(self,l):
        self.__l=l

    def get_side(self):
        return self.__l

    def set_side(self,new_side):
        if new_side > 0:
            self.__l=new_side
        else:
            print('Side length must be positive!')

squ=Square(5)
print(f'initial length is : {squ.get_side()}')
squ.set_side(10)
print(f'updated side is : {squ.get_side()}')
squ.set_side(-1)
```

```
initial length is : 5
updated side is : 10
Side length must be positive!
```

14) Create a class Profit that has a method named getProfit that accepts profit from the user.

Create a class Loss that has a method named getLoss that accepts loss from the user.

Create a class BalanceSheet that inherits from both classes Profit and Loss and calculates the balanace. It has two methods getBalance() and printBalance().

In [30]:
```
class Profit:
    def __init__(self):
        self.profit=0

    def getProfit(self):
        self.profit=int(input('Enter profit : '))

class Loss:
    def __init__(self):
        self.loss=0

    def getLoss(self):
        self.loss=int(input('Enter loss : '))

class BalanceSheet(Profit,Loss):
    def __init__(self):
        super().__init__()
        Profit.__init__(self)
        Loss.__init__(self)

    def getBalance(self):
```

```
        return self.profit - self.loss

    def printBalance(self):
        print(f'Balance is : {self.getBalance()}')


b1=BalanceSheet()
b1.getProfit()
b1.getLoss()
b1.printBalance()
```

```
Balance is : 9980
```

15) WAP to demonstrate all types of inheritance.

In [38]:
```
# 1 : SINGLE INHERITANCE
class Animal:
    def sound(self):
        print('Animals make sound')

class Dog(Animal):
    def bark(self):
        print('Dogs bark')

d=Dog()
d.sound()
d.bark()

# 2 : MULTIPLE INHERITANCE
class Flyable:
    def fly(self):
        print("Can fly.")

class Bird(Animal, Flyable):
    def chirp(self):
        print("Bird chirps.")

bird = Bird()
bird.sound()
bird.fly()
bird.chirp()

# 3. Multilevel Inheritance (Grandparent → Parent → Child)
class Mammal(Animal):  # Animal → Mammal → Human
    def has_fur(self):
        print("Most mammals have fur.")

class Human(Mammal):
    def speak(self):
        print("Humans can speak.")

human = Human()
human.sound()
human.has_fur()
human.speak()


# 4. Hierarchical Inheritance (One parent, multiple children)
class Cat(Animal):
```

```
    def meow(self):
        print("Cat meows.")

cat = Cat()
cat.sound()
cat.meow()


# 5. Hybrid Inheritance (Combination of Multiple and Multilevel)
class Bat(Mammal, Flyable):
    def night_hunt(self):
        print("Bat hunts at night.")

bat = Bat()
bat.sound()
bat.has_fur()
bat.fly()
bat.night_hunt()
```

```
Animals make sound
Dogs bark
Animals make sound
Can fly.
Bird chirps.
Animals make sound
Most mammals have fur.
Humans can speak.
Animals make sound
Cat meows.
Animals make sound
Most mammals have fur.
Can fly.
Bat hunts at night.
```

16) Create a Person class with a constructor that takes two arguments name and age.

Create a child class Employee that inherits from Person and adds a new attribute salary.

Override the **init** method in Employee to call the parent class's **init** method using the super() and then initialize the salary attribute.

In [42]:
```
class Person:
    def __init__(self,name,age):
        self.name=name
        self.age=age

    def display(self):
        print(f'Person\'s name is {self.name} and age is {self.age}')

class Employee(Person):
    def __init__(self,name,age,salary):
        super().__init__(name,age)
        self.salary=salary
```

```
    def display(self):
        super().display()
        print(f'and salary is {self.salary}')

e1=Employee('Tanisha',19,100000)
e1.display()
```

```
Person's name is Tanisha and age is 19
and salary is 100000
```

17) Create a Shape class with a draw method that is not implemented.

Create three child classes Rectangle, Circle, and Triangle that implement the draw method with their respective drawing behaviors.

Create a list of Shape objects that includes one instance of each child class, and then iterate through the list and call the draw method on each object.

In [50]:
```
class Shape:
    def draw(self):
        pass

class Rectangle(Shape):
    def draw(self):
        print('Shape is Rectangle')


class Circle(Shape):
    def draw(self):
        print('Shape is Circle')



class Triangle(Shape):
    def draw(self):
        print('Shape is Triangle')

r1=Rectangle()
r1.draw()
c1=Circle()
c1.draw()
t1=Triangle()
t1.draw()
```

```
Shape is Rectangle
Shape is Circle
Shape is Triangle
```

In [ ]:
```

```