Python Programming - 2301CS404 Lab - 1 Shruti | 23010101311 | 25-11-2024

01) WAP to print “Hello World.

```
print('Hello World')

Hello World
```

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

```
# With Using Input
a = int(input('Enter First Number :'))
b = int(input('Enter Second Number :'))
print(a,'+',b,'=',(a+b))

Enter First Number : 5
Enter Second Number : 6

5 + 6 = 11

# Without Using Input
a = 10
b = 5
sum = a + b
print(a,'+',b,'=',sum)

10 + 5 = 15
```

03) WAP to check the type of the variable.

```
a = 'Shruti'
print(type(a))

<class 'str'>
```

04) WAP to calculate simple interest.

```
p = float(input('Enter P : '))
r = float(input('Enter R : '))
n = float(input('Enter N : '))
I = (p*r*n)/100
print('Simple Interest =',I)

Enter P :  15
Enter R :  10
Enter N :  60.4

Simple Interest = 90.6
```

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

```
r = float(input('Enter a radius of a circle : '))
Area = 3.14*r*r
Perimeter = 2*3.14*r
print('Area of a Circle =',Area)
print('Perimeter of a Circle =',Perimeter)

Enter a radius of a circle :  3

Area of a Circle = 28.259999999999998
Perimeter of a Circle = 18.84
```

06) WAP to calculate area of a triangle.

```
h = float(input('Enter height of a triangle :'))
b = float(input('Enter breadth of a triangle :'))
print('Area of a Triangle =',(0.5*h*b))

Enter height of a triangle : 4
Enter breadth of a triangle : 9

Area of a Triangle = 18.0
```

07) WAP to compute quotient and remainder.

```
num1 = float(input('Enter num1 :'))
num2 = float(input('Enter num2 :'))
q = num1 / num2
r = num1 % num2
print('Quotient =',q,'\nRemainder =',r)

Enter num1 : 10
Enter num2 : 2

Quotient = 5.0
Remainder = 0.0
```

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

```
c = float(input('Enter temp in celsius:'))
f = float(input('Enter temp in fahrenheit:'))
F = c * (9/5) + 32
C = (f - 32) * 5/9
print(f,'Fahrenheit =',C,'Degree Celsius')
print(c,'Degree Celsius =',F,'Fahrenheit')

Enter temp in celsius: 23
Enter temp in fahrenheit: 73.4

73.4 Fahrenheit = 23.000000000000004 Celsius
23.0 Celsius = 73.4 Fahrenheit
```

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

```python
import math
p11 = int(input('Enter x of point1 :'))
p12 = int(input('Enter y of point1 :'))
p21 = int(input('Enter x of point2 :'))
p22 = int(input('Enter y of point2 :'))
d1 = (p12-p11)
d2 = (p22-p21)
dis = math.sqrt(d1*d1 + d2*d2)
print('Distance of this two points is =',dis)
```

```
Enter x of point1 : 1
Enter y of point1 : 2
Enter x of point2 : 2
Enter y of point2 : 4

Distance of this two points is = 2.23606797749979
```

10) WAP to print sum of n natural numbers.

```python
n = int(input('Enter Number :'))
sum = 0
for i in range(1,n+1):
    sum = sum + i
print('Sum of',n,'natural numbers is =',sum)
```

```
Enter Number : 5

Sum of 5 natural numbers is = 15
```

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

```python
n = int(input('Enter Number :'))
sum = 0
for i in range(1,n+1):
    sum = sum + (i*i)
print('Sum of square of',n,'natural numbers is =',sum)
```

```
Enter Number : 4

Sum of square of 4 natural numbers is = 30
```

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

```python
F = input('Enter the first name of the student :')
L = input('Enter the last name of the student :')
Name = F + ' ' + L
print('Student Name = ',Name)
```

```
Enter the first name of the student : Zalariya
Enter the last name of the student : Shruti

Student Name =  Zalariya Shruti
```

```python
F = input('Enter the first name of the student :')
L = input('Enter the last name of the student :')
print('Student Name :',F,L)
```

```
Enter the first name of the student : Ambani
Enter the last name of the student : Shruti

Student Name : Ambani Shruti
```

13) WAP to swap two numbers.

```python
a = int(input('Enter first number :'))
b = int(input('Enter second number :'))
print('Before Swap!')
print('a =',a,' b =',b)
a = a + b
b = a - b
a = a - b
print('After Swap!')
print('a =',a,' b =',b)
```

```
Enter first number : 5
Enter second number : 10

Before Swap!
a = 5   b = 10
After Swap!
a = 10   b = 5
```

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

```python
d = float(input('Enter distance into kilometer :'))
m = 1000*d
f = 3280.84*d
i = 39370.1*d
cm = 100000*d
print(d,'km =',m,'meters')
print(d,'km =',f,'feets')
print(d,'km =',i,'inches')
print(d,'km =',cm,'centimeters')
```

```
Enter distance into kilometer : 10

10.0 km = 10000.0 meters
10.0 km = 32808.4 feets
```

```
10.0 km = 393701.0 inches
10.0 km = 1000000.0 centimeters
```

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

```
D = input('Enter the day :')
M = input('Enter the month :')
Y = input('Enter the year :')
Date = D + '-' + M + '-' + Y
print('Date :',Date)
```

```
Enter the day : 23
Enter the month : 11
Enter the year : 2024

Date : 23-11-2024
```

python-programming-lab-2

March 11, 2025

Python Programming - 2301CS404

Lab - 2

Shruti | 23010101311 | 02-12-2024

1 if..else..

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

[5]:
```
a = int(input('Enter a number : '))

if(a>0):
    print(a ,'is positive.')
else:
    print(a ,'is negative')
```

```
Enter a number :  5
```

```
5 is positive.
```

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

[7]:
```
a = int (input('Enter a number : '))

if(a%2 == 0):
    print(a ,'is Even.')
else:
    print(a ,'is Odd.')
```

```
Enter a number :  10
```

```
10 is Even.
```

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

[9]:
```
# Simple If

a = int (input('Enter a number : '))
b = int (input('Enter a number : '))

if(a>b):
    print(a ,'is Largest number.')
else:
    print(b ,'is Largest number.')
```

```
Enter a number :  5
Enter a number :  10

10 is Largest number.
```

[11]:
```
# Ternary Operator

a = int (input('Enter a number : '))
b = int (input('Enter a number : '))

print(a , 'is largest number.') if(a>b) else print(b , 'is largest number.')
```

```
Enter a number :  5
Enter a number :  3

5 is largest number.
```

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

[15]:
```
# Simple If_elseif

a = int (input('Enter a first number a : '))
b = int (input('Enter a second number b : '))
c = int (input('Enter a third number c : '))

if(a>b):
    if(a>c):
        print(a ,'is Largest number.')
    else:
        print(c ,'is Largest number.')
else:
    if(b>c):
        print(b ,'is Largest number.')
    else:
        print(c ,'is Largest number.')
```

```
Enter a first number a :  10
Enter a second number b :  13
Enter a third number c :  4
```

```
13 is Largest number.
```

[21]:
```python
# Ternary Operator

a = int (input('Enter a first number a : '))
b = int (input('Enter a second number b : '))
c = int (input('Enter a third number c : '))

print(a,'is Largest number.') if a>b and a>c else print(b,'is Largest number.')␣
 ↪if b>c else print(c,'is Largest number.')
```

```
Enter a first number a :  12
Enter a second number b :  13
Enter a third number c :  15
```

```
15 is Largest number.
```

1.0.5 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]

[23]:
```python
a = int (input('Enter a Year : '))

if(a%4==0):
    if(a%100!=0):
        print(a ,'is Leap Year.')
    else:
        print(a ,'is not Leap Year.')
elif(a%400==0):
    print(a ,'is Leap Year.')
else:
    print(a ,'is not Leap Year.')
```

```
Enter a Year :  2024
```

```
2024 is Leap Year.
```

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

[25]:
```python
#Simple elif
d = int(input('Enter a number between 0 to 6 : '))

if(d==0):
    print('Sunday')
```

```
elif(d==1):
    print('Monday')
elif(d==2):
    print('Tuesday')
elif(d==3):
    print('Wednesday')
elif(d==4):
    print('Thursday')
elif(d==5):
    print('Friday')
elif(d==6):
    print('Saturday')
else:
    print('Please Enter A valid Number')
```

```
Enter a number between 0 to 6 :  2
```

```
Tuesday
```

[27]:
```
# Match case
d = int(input('Enter a number between 0 to 6 : '))

match d:
    case 0:
        print('Sunday')
    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 _:
        print('Please Enter A valid Number')
```

```
Enter a number between 0 to 6 :  3
```

```
Wednesday
```

[29]:
```
#Simple elif
d = int(input('Enter a number : '))

if(d%7==0):
    print('Sunday')
elif(d%7==1):
    print('Monday')
elif(d%7==2):
    print('Tuesday')
elif(d%7==3):
    print('Wednesday')
elif(d%7==4):
    print('Thursday')
elif(d%7==5):
    print('Friday')
elif(d%7==6):
    print('Saturday')
else:
    print('Please Enter A valid Number')
```

```
Enter a number :  15
```

```
Monday
```

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

[37]:
```
a = int (input('Enter a first number a : '))
b = int (input('Enter a second number b : '))
c = int (input('Enter 1 for Addition , Enter 2 for Subtraction , Enter 3 for␣
 ↪Multiplication , Enter 4 for Division :'))

if(c==1):
    print(a,'+',b,'=', a+b)
elif(c==2):
    print(a,'-',b,'=', a-b)
elif(c==3):
    print(a,'*',b,'=', a*b)
elif(c==4):
    print(a,'/',b,'=', a/b)
else:
    print('Please enter valid operator')
```

```
Enter a first number a :  6
Enter a second number b :  3
Enter 1 for Addition , Enter 2 for Subtraction , Enter 3 for Multiplication ,
Enter 4 for Division : 4
```

```
6 / 3 = 2.0
```

1.0.8 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

[41]:
```python
a = int (input('Enter a first number a : '))
b = int (input('Enter a second number b : '))
c = int (input('Enter a third number c : '))
d = int (input('Enter a fourth number d : '))
e = int (input('Enter a fifth number e : '))
Total = a+b+c+d+e
print ('Total marks = ' , Total)
Percentage = (Total*100)/500
print ('Percentage = ' , Percentage)

if(Percentage>70):
    print('Distinction')
elif(Percentage>=60):
    if(Percentage<=70):
        print('First Class')
elif(Percentage>=45):
    if(Percentage<60):
        print('Second Class')
elif(Percentage>=35 or Percentage<45):
    print('Pass')
if(Percentage<35):
    print('Fail')
```

```
Enter a first number a :  50
Enter a second number b :  100
Enter a third number c :  50
Enter a fourth number d :  50
Enter a fifth number e :  50
```

```
Total marks =  300
Percentage =  60.0
First Class
```

1.0.9 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.

[13]:
```python
a = float(input("Enter the first side: "))
b = float(input("Enter the second side: "))
c = float(input("Enter the third side: "))

if a + b > c and b + c > a and a + c > b:
    # Check for equilateral triangle
    if a == b == c:
        print("The triangle is Equilateral.")
    # Check for isosceles triangle
    elif a == b or b == c or a == c:
        print("The triangle is Isosceles.")
    # If none of the above, it's a scalene triangle
    elif a**2 + b**2 == c**2 or b**2 + c**2 == a**2 or a**2 + c**2 == b**2 and␣
 ↪a!=b!=c:
        print("The triangle is Right-Angled and Scalene.")
    # Check for right-angled triangle using Pythagoras theorem
    elif a**2 + b**2 == c**2 or b**2 + c**2 == a**2 or a**2 + c**2 == b**2:
        print("The triangle is Right-Angled.")
else:
    print("The sides do not form a valid triangle.")
```

```
Enter the first side:  3
Enter the second side:  4
Enter the third side:  5

The triangle is Right-Angled and Scalene.
```

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

[45]:
```
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))
num3 = float(input("Enter the third number: "))

if (num1 >= num2 and num1 <= num3) or (num1 <= num2 and num1 >= num3):
    second_largest = num1
elif (num2 >= num1 and num2 <= num3) or (num2 <= num1 and num2 >= num3):
    second_largest = num2
else:
    second_largest = num3

print("The second largest number is:", second_largest)
```

```
Enter the first number:  5
Enter the second number:  4
Enter the third number:  9

The second largest number is: 5.0
```

1.0.11 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

[47]:
```
units = float(input("Enter the number of units consumed: "))

if units>0 or 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("The total electricity bill is: Rs.",bill)
```

```
Enter the number of units consumed:  55

The total electricity bill is: Rs. 143.0
```

[ ]:
```

```

for and while loop

01) WAP to print 1 to 10.

```
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.

```
n = int(input('Enter the number of iteration :'))
for i in range(1,n+1,1):
    print(i)

Enter the number of iteration : 5

1
2
3
4
5
```

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

```
n = int(input('Enter the number of iteration :'))
for i in range(1,n+1,2):
    print(i)

Enter the number of iteration : 8

1
3
5
7
```

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

```
n = int(input('Enter first number :'))
m = int(input('Enter second number :'))
for i in range(n,m+1):
    if(i%2==0):
        if(i%3!=0):
            print(i)
```

```
Enter first number : 2
Enter second number : 12

2
4
8
10
```

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

```
n = int(input('Enter number :'))
sum = 0
for i in range(1,n+1,1):
    sum = sum + i
print('Sum =',sum)
```

```
Enter first number : 10

Sum = 55
```

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

```
n = int(input('Enter number :'))
sum = 0
for i in range(1,n+1):
    sum = sum + (i*i)
print('Sum =',sum)
```

```
Enter first number : 4

Sum = 30
```

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

```
n = int(input('Enter first number :'))
sum = 0
for i in range(1,n+1):
    if(i%2==0):
        sum-=i
    else:
        sum+=i
```

```
else:
    print('Sum =',sum)

Enter first number : 3

Sum = 2
```

08) WAP to print multiplication table of given number.

```
n = int(input('Enter number :'))
ans = 0
for i in range(1,11):
    ans = n*i
    print(n,'*',i,'=',ans)

Enter number : 5

5 * 1 = 5
5 * 2 = 10
5 * 3 = 15
5 * 4 = 20
5 * 5 = 25
5 * 6 = 30
5 * 7 = 35
5 * 8 = 40
5 * 9 = 45
5 * 10 = 50
```

09) WAP to find factorial of the given number.

```
n = int(input('Enter number :'))
ans = 1
for i in range(1,n+1):
    ans = ans*i
else:
    print('Factorial of number',n,'=',ans)

Enter number : 5

Factorial of number 5 = 120
```

10) WAP to find factors of the given number.

```
n = int(input('Enter number :'))
print('Factor of number',n,':')
for i in range(1,n+1):
    if n%i==0:
        print(i)

Enter number : 10
```

```
Factor of number 10 :
1
2
5
10
```

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

```
n = int(input('Enter number :'))
count = 0
for i in range(2,int(n/2)+1):
    if n%i==0:
        count += 1;
        break;
if count==0:
    print('Number',n,'is Prime number!')
else:
    print('Number',n,'is not prime number!')


Enter number : 4

Number 4 is not prime number!
```

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

```
n = int(input('Enter number :'))
r = 0
sum = 0
while n>0:
    r = n%10
    sum = sum+r
    n = int(n/10)
else:
    print('Sum =',sum)

Enter number : 123

Sum = 6
```

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

```
n = int(input('Enter number :'))
r = 0
ans = 0
num = n
while n>0:
    r = n%10
    ans = (ans*10)+r
    n = int(n/10)
    #n = n//10
print('reverse =',ans)
if ans == num:
    print(num,'is palindrom number!')
else:
    print(num,'is not palindrom number!')
```

```
Enter number : 161

reverse = 161
161 is palindrom number!
```

14) WAP to print GCD of given two numbers.

```
n = int(input('Enter first number :'))
m = int(input('Enter second number :'))
min = n
gcd = 0
if n>m:
    min=m
for i in range(1,min+1):
    if(n%i==0 and m%i==0):
        gcd = i
print('GCD of',n,'and',m,'=',gcd)
```

```
Enter first number : 5
Enter second number : 15

GCD of 5 and 15 = 5
```

String

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

```
str = input('Enter a string :')
str1 = str[::-1]
if(str==str1):
    print(str,'is palindrom string.')
else:
    print(str,'is not palindrom string.')
```

```
Enter a string : abcdcba

abcdcba is palindrom string.
```

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

```
str = input('Enter a string :')
print('Original string is :',str)
str1 = str.split(' ')
str2 = str1[::-1]
str3 = ' '.join(str2)
print('Reverse word string is :',str3)
```

```
Enter a string : zalariya shruti b.tech

Original string is : zalariya shruti b.tech
Reverse word string is : b.tech shruti zalariya
```

03) WAP to remove ith character from given string.

```
str = input('Enter a string :')
a = int(input('Enter index for remove character :'))
for i in str:
    if i!=a:
        print(i);
    else:
        print();
```

```
Enter a string : shruti
Enter index for remove character : 3

s
h
r
```

```
u
t
i
```

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

```
str = input('Enter a string :')
len = 0
for i in str:
    len+=1
print('Length of this string is =',len)
```

```
Enter a string : zalariya

Length of this string is = 8
```

05) WAP to print even length word in string.

```
str = input('Enter a string :')
str1 = str.split(' ')
for i in str1:
    l = len(i)
    if l%2==0:
        print(i)
```

```
Enter a string : shruti riya nandu

shruti
riya
```

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

```
str = input('Enter a string :')
count = 0
for i in str:
    if 'a' in i or 'e' in i or 'i' in i or 'o' in i or 'u' in i:
        count+=1
print('Number of vowels in string =',count)
```

```
Enter a string : darshan university

Number of vowels in string = 6
```

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

```
str = input('Enter string: ')

str1 = str.split(' ')
```

```
str2 = []

for i in str1:
    if len(i) > 1:
        i = i[0].upper() + i[1:len(i)-1] + i[-1].upper()
    elif len(i) == 1:
        i = i.upper()

    # Add the modified word to the list
    str2.append(i)

# Join the modified words into a single string
result_str = ' '.join(str2)

# Print the final result
print("Modified string:", result_str)
```

```
Enter string:  Zalariya shrutI r

Modified string: ZalariyA ShrutI R
```

08) WAP to convert given array to string.

```
print('Enter Array Elements!')
for i in range(0,6):
    element = input(f'a[{i}] =')
    a.append(element)
s = ''
s.join(a)
```

```
Enter Array Elements!

a[0] = s
a[1] = h
a[2] = r
a[3] = u
a[4] = t
a[5] = i

'shruti'
```

```
a = ['M','o','r','b','i']
''.join(a)
```

```
'Morbi'
```

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

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

```
Pass = input('Enter Password :')
c_pass = input('Enter Confirm Password :')
if Pass==c_pass:
    print('Both are Same!')
else:
    if(Pass.lower()==c_pass.lower()):
        print('Cases mistake are there!')
    else:
        print('Both are not same!\nSomething went wrong!!!!!')
```

```
Enter Password : Shruti
Enter Confirm Password : SHRUTI

Cases mistake are there!
```

10) : Display credit card number.

card no. : 1234 5678 9012 3456

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

```
card_no = '1234 5678 9012 3456'
a = card_no.split(' ')
for i in a:
    card = i[:12].replace(i[:12],'**** **** **** ') + i[-4:]
print(card)
```

```
**** **** **** 3456
```

```
card_no = '1234 5678 9012 3456'
a = card_no.split(' ')
card = '**** **** **** ' + a[3]
print(card)
```

```
**** **** **** 3456
```

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

s1 = decimal and s2 = medical are Anagram

```
s1 = input('Enter str 1: ')
s2 = input('Enter str 2: ')

# Convert both strings to lowercase
s1 = s1.lower()
s2 = s2.lower()
```

```
# Sort both strings
s3 = ''.join(sorted(s1))
s4 = ''.join(sorted(s2))

if s3 == s4:
    print('Strings are an Anagram!')
else:
    print('Strings are not an Anagram!')
```

```
Enter str 1:  ieieieie
Enter str 2:  ieieieee

Strings are not an Anagram!
```

12) : Rearrange the given string. First lowercase then uppercase alphabets.

input : EHlsarwiwhtwMV

output : lsarwiwhtwEHMV

```
str = input('Enter a string:')
str1 = ''
str2 = ''
for i in str:
    if i.isupper():
        str1+=i
    else :
        str2+=i
str3 = str2 + str1
print(str3)
```

```
Enter a string: EHlsarwiwhtwMV

lsarwiwhtwEHMV
```

List

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

```
l1 = [1,2,3,4,5]
sum = 0
for i in l1:
    sum = sum + i
print('Sum of all elements is =',sum)
```

```
Sum of all elements is = 15
```

02) WAP to find largest element in a List.

```
l1 = [1,2,3,4,5,10,8,15,2]
max = 0
for i in l1:
    if(i > max):
        max = i
print('Maximum number =',max)
```

```
Maximum number = 15
```

03) WAP to find the length of a List.

```
l1 = [1,2,3,4,5,10,8,15,2]
length = len(l1)
print('Length of the list =',length)
```

```
Length of the list = 9
```

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

```
l1 = [1,2,3,4,5,10,8,15,2,10]
temp = l1[0]
l1[0] = l1[-1]
l1[-1] = temp
print(l1)
```

```
[10, 2, 3, 4, 5, 10, 8, 15, 2, 1]
```

```
l1 = [1,2,3,4,5]
l1[0] , l1[len(l1)-1] = l1[len(l1)-1] , l1[0]
l1
```

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

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

```
li=[1,2,3,4,5,6,7,8,9,10]
mid = len(li)//2
l1 = li[:mid:]
l2 = li[mid::]
l = l2 + l1
print(l)
```

```
[6, 7, 8, 9, 10, 1, 2, 3, 4, 5]
```

```
li=[1,2,3,4,5,6,7,8,9,10]
mid = len(li)//2
l1 = li[:mid:]
l2 = li[mid::]
l2.extend(l1)
print(l2)
```

```
[6, 7, 8, 9, 10, 1, 2, 3, 4, 5]
```

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

```
l1 = []
for i in range(1,6,1):
    l1.append(int(input('Enter list elements : ')))
print('Before swapping!')
print(l1)

a = int(input('Enter first index :'))
b = int(input('Enter second index :'))
temp = l1[a]
l1[a] = l1[b]
l1[b] = temp

print('After swapping!')
print(l1)
```

```
Enter list elements :  1
Enter list elements :  2
Enter list elements :  3
Enter list elements :  4
Enter list elements :  5

Before swapping!
[1, 2, 3, 4, 5]

Enter first index : 1
Enter second index : 3
```

```
After swapping!
[1, 4, 3, 2, 5]
```

07) WAP to reverse the list entered by user.

```
l1 = []
for i in range(1,6,1):
    l1.append(int(input('Enter list elements : ')))

print('Reverse list!')
print(l1[::-1])
```

```
Enter list elements :  1
Enter list elements :  2
Enter list elements :  3
Enter list elements :  4
Enter list elements :  5

Reverse list!
[5, 4, 3, 2, 1]
```

08) WAP to print even numbers in a list.

```
l1 = []
for i in range(1,6,1):
    l1.append(int(input('Enter list elements : ')))

for i in l1:
    if i%2==0:
        print(i)
```

```
Enter list elements :  1
Enter list elements :  23
Enter list elements :  5
Enter list elements :  44
Enter list elements :  22

44
22
```

09) WAP to count unique items in a list.

```
l1 = [1,2,3,1,5,6,7]
l2 = set(l1)
print(l2)
count = 0
for i in l2:
    count += 1
print('Count of unique items =',count)
```

```
{1, 2, 3, 5, 6, 7}
Count of unique items = 6
```

```python
l1 = [1,2,3,1,5,6,7]
count=0
for i in l1:
    if l1.count(i)==1:
        print(i)
        count+=1
print('Count of unique items =',count)
```

```
2
3
5
6
7
Count of unique items = 5
```

10) WAP to copy a list.

```python
l1 = [1,2,3,1,5,6,7,8]
l1.copy()
```

```
[1, 2, 3, 1, 5, 6, 7, 8]
```

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

```python
n = int(input('Enter range :'))
l1 = [i for i in range(0,(n+1)) if i%2!=0]
print(l1)
```

```
Enter range : 10

[1, 3, 5, 7, 9]
```

```python
n = int(input('Enter range :'))
l1 = [i for i in range(1,(n+1),2)]
print(l1)
```

```
Enter range : 10

[1, 3, 5, 7, 9]
```

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

```python
l1 = [1,2,3,3,3,5,6,7,8]
n = int(input('Enter Element :'))
count = 0
for i in l1:
    if n==i:
        count+=1
print('Occurence of number 3 =',count)

Enter Element : 3

Occurence of number 3 = 3

l1 = [1,2,3,3,3,5,6,7,8]
n = int(input('Enter Element :'))
print('Occurence of number 3 =',l1.count(n))

Enter Element : 3

Occurence of number 3 = 3
```

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

```
l1 = []
print('Enter list elements!')
for i in range(0,10,1):
    l1.append(int(input(f'l1[{i}] =')))

l1 = list(set(l1))
l1.sort(reverse=True)
print(l1)
if len(l1) > 1:
    print('Second largest number is',l1[1])
else:
    print('There is no second largest number.')

Enter list elements!

l1[0] = 1
l1[1] = 4
l1[2] = 0
l1[3] = 20
l1[4] = 70
l1[5] = 5
l1[6] = 3
l1[7] = 8
l1[8] = 6
l1[9] = 4

[70, 20, 8, 6, 5, 4, 3, 1, 0]
Second largest number is 20
```

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

```
l1 = [1, 2, 3, 3, 3, 5, 6, 7, 8, 6, 3, 5, 5, 6]
k = int(input('Enter frequency: '))
```

```
ans = []

for i in l1:
    count = 0
    for j in l1:
        if i == j:
            count += 1

    if count > k and i not in ans:
        ans.append(i)

print(ans)

Enter frequency:  2

[3, 5, 6]

l1 = [1, 2, 3, 3, 3, 5, 6, 7, 8, 6, 3, 5, 5, 6]
k = int(input('Enter frequency :'))
f = {}
for i in l1:
    if i in f:
        f[i]+=1
    else:
        f[i] = 1
ans = [i for i,count in f.items() if count > k ]
print(ans)

Enter frequency : 3

[3]
```

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

```
#Withou Using Comprehension
l1 = []
for i in range(0,10):
    l1.append(i**2)
print(l1)

[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]

#With Using Comprehension
l1 = [i**2 for i in range(0,10)]
print(l1)

[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.

```
l1 = 
['Banana','blueBerry','Apple','Kiwi','blackBerry','Cherry','Brazil 
Nut','Chickoo']
for i in l1:
    if i[0]=='B' or i[0]=='b':
        l2.append(i)
print(l2)
```

```
['Banana', 'blueBerry', 'blackBerry', 'Brazil Nut']
```

```
l1 = 
['Banana','BlueBerry','Apple','Kiwi','BlackBerry','Cherry','Brazil 
Nut','Chickoo']
l2 = [i for i in l1 if i[0]=='B' or i[0]=='b']
print(l2)
```

```
['Banana', 'BlueBerry', 'BlackBerry', 'Brazil Nut']
```

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

```
l1 = [1,2,3,4,5]
l2 = [5,3,8,9,1]
l3 = []
for i in l1:
    for j in l2:
        if i == j:
            l3.append(i)
print('Common elements are :',l3)
```

```
Common elements are : [1, 3, 5]
```

```
l1 = [1,2,3,4,5]
l2 = [5,3,8,9,1]
l3 = []
for i in l1:
    if i in l2:
        l3.append(i)
print('Common elements are :',l3)
```

```
Common elements are : [1, 3, 5]
```

```
l1 = [1,2,3,4,5]
l2 = [5,3,8,9,1]
l3 = [i for i in l1 if i in l2]
print('Common elements are :',l3)
```

```
Common elements are : [1, 3, 5]
```

Tuple

01) WAP to find sum of tuple elements.

```
t1 = (1,2,3,4,5)
sum = 0
for i in t1:
    sum = sum + i;
print('Sum =',sum)
```

```
Sum = 15
```

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

```
t1 = (10, 20, 30, 40, 50, 1, 2, 3, 4, 5)
k = int(input('Enter K :'))
t1 = sorted(t1) #Sorted attribute sort the data
min = t1[:k] #strat from index 0 upto k-1
max = t1[-k:] #strat from -k end to -1 and last index is always -1 in
reverse case
print('Max k elements are :',max,'\nMin k elements are :',min)
```

```
Enter K : 3

Max k elements are : [30, 40, 50]
Min k elements are : [1, 2, 3]
```

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

```
t1 = (10,20,30,40,50)
t2 = (1,2,3,4,5)
t3 = (2,4,6,8,10)

l1 = [t1,t2,t3]
k = int(input('Enter k :'))

for i in l1:
    count = 0
    for j in i:
        if j % k == 0:
            count+=1
    if(count==len(i)):
        print(i)
```

```
Enter k : 2

(10, 20, 30, 40, 50)
(2, 4, 6, 8, 10)
```

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

```
l1 = [2,4,6,8,10]
ans = [(i,i**3) for i in l1]
print(ans)

[(2, 8), (4, 64), (6, 216), (8, 512), (10, 1000)]
```

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

```
t1 = (10,20,30,-40,50)
t2 = (1,2,-3,4,5)
t3 = (2,4,6,8,10)
l1 = [t1,t2,t3]

for i in l1:
    count = 0
    for j in i:
        if j>0:
            count+=1
    if(count==len(i)):
        print(i)

(2, 4, 6, 8, 10)
```

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

```
t1 = (1,2)
t2 = (3,4)
t3 = (10,20)

l1 = [t1,t2]

l2 = [5,6]
l3 = [7,8]

t4 = (l1,l2)

l1.append(t3)
print(l1)

t4 = list(t4)
t4.append(l3)
```

```
t4 = tuple(t4)
t4
```

```
[(1, 2), (3, 4), (10, 20)]
```

```
([[(1, 2), (3, 4), (10, 20)], [5, 6], [7, 8])
```

```
t1 = (1,2)
t2 = (3,4)
t3 = (10,20)

l1 = [t1,t2]

l2 = [5,6]
l3 = [7,8]

t4 = (l1,l2)

l1.append(t3)
print(l1)

t4 = t4 + (l3,)
print(t4)
```

```
[(1, 2), (3, 4), (10, 20)]
([[(1, 2), (3, 4), (10, 20)], [5, 6], [7, 8])
```

07) WAP to remove tuples of length K.

```
t1 = (1,2,5,6,7)
t2 = (3,4)
t3 = (10,20,30)
t4 = (1,2,3,4,5,6)

l1 = [t1,t2,t3,t4]
l2 = []

k = int(input('Enter length of tuple :'))

for i in l1:
    if len(i)!=k:
        l2.append(i)
print(l2)
```

```
Enter length of tuple : 6
```

```
[(1, 2, 5, 6, 7), (3, 4), (10, 20, 30)]
```

08) WAP to remove duplicates from tuple.

```
t1 = (1, 2, 2, 3, 3, 4, 4, 4, 5, 6, 7, 8)
t2 = tuple(set(t1))
print(t2)
```

```
(1, 2, 3, 4, 5, 6, 7, 8)
```

```
t1 = (1, 2, 2, 3, 3, 4, 4, 4, 5, 6, 7, 8)
t2 = ()
for i in t1:
    if i not in t2:
        t2 = t2 + (i,)
print(t2)
```

```
(1, 2, 3, 4, 5, 6, 7, 8)
```

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

```
t1 = (1, 2, 3, 4, 5)
t2 = ()
for i in range(len(t1)-1):
    m = t1[i]*t1[i+1]
    t2 = t2 + (m,)
print(t2)
```

```
(2, 6, 12, 20)
```

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

```
t1 = (1,2,3,4,5)
for i in t1:
    if(t1.count(i)!=1):
       print('Tuple has duplicate members.')
       break;
else:
     print('Tuple is distinct.')
```

```
Tuple is distinct.
```

Set & Dictionary

01) WAP to iterate over a set.

```
s1 = {1,2,3,4,5}
for i in s1:
    print(i)
```

```
1
2
3
4
5
```

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

```
s1 = {1,2,3,4,5}
print('Set :',s1)
l1 = list(s1)
print('List :',l1)
print(type(l1))
s = str(s1)
print('String :',s)
print(type(s))
t1 = tuple(s1)
print('Tuple :',t1)
print(type(t1))
```

```
Set : {1, 2, 3, 4, 5}
List : [1, 2, 3, 4, 5]
<class 'list'>
String : {1, 2, 3, 4, 5}
<class 'str'>
Tuple : (1, 2, 3, 4, 5)
<class 'tuple'>
```

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

```
s1 = {0,1,2,3,4,5}
max = i
min = i
for i in s1:
        if i < min:
            min = i
        if i > max:
            max = i
print('Maximum num :',max)
print('Minimum num :',min)
```

```
Maximum num : 5
Minimum num : 0
```

04) WAP to perform union of two sets.

```
s1 = {0,1,2,3,4}
s2 = {5,6,7,8,9}
s1.union(s2)
#{0,1,2,3,4}.union({5,6,7,8,9})
```

```
{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
```

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

```
l1 = [1,2,3,4,5]
l2 = [6,7,8,5,4,2,9]
count = 0
for i in l1:
    if i in l2:
        count+=1
if count!=0:
    print('Yes, two lists have',count,'common elements.')
else:
    print('No, two lists have not common elements.')
```

```
Yes, two lists have 3 common elements.
```

06) WAP to remove duplicates from list.

```
l1 = [1,2,1,2,6,7,8,9]
l2 = []
for i in l1:
    if i not in l2:
        l2.append(i)
print(l2)
```

```
[1, 2, 6, 7, 8, 9]
```

```
l1 = [1,2,1,2,6,7,8,9]
l1 = list(set(l1))
l1
```

```
[1, 2, 6, 7, 8, 9]
```

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

```
str = input('Enter String :')
str1 = set(str.split(' '))
print(str1)

Enter String : hi hello how hi are are

{'hi', 'are', 'hello', 'how'}
```

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

```
a = {1,2,3,4,5}
b = {4,5,6,7,8}
a.difference(b)

{1, 2, 3}
```

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

```
s1 = 'shruti'
s2 = 'hruitss'
st1 = set(s1)
st2 = set(s2)
for i in st1:
    if i not in st2 or len(s1)!=len(s2):
        print('Given string is not anagram!')
        break
else:
    print('Given string is anagram!')

Given string is not anagram!
```

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

```
l1 = [1,2,3]
l2 = [2,4,5]
l3 = [3,2,5]
s1 = set(l1)
s2 = set(l2)
s3 = set(l3)
s1 & s2 & s3

{2}
```

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

```
str = input('Enter String:')
s1 = {'a','e','i','o','u','A','E','I','O','U'}
count = 0
```

```
for i in str:
    if i in s1:
        count+=1
print('Num of vowels in given string =',count)

Enter String: how are yOu

Num of vowels in given string = 5
```

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

```
str = input('Enter String:')
count = 0
for i in str:
    if i=='0' or i=='1':
        count+=1
if len(str)==count:
    print('Given string is Binary!')
else:
     print('Given string is not Binary!')

Enter String: 0101

Given string is Binary!

str = input('Enter String:')
s1 = {'0','1'}
for i in str:
    if i not in s1:
        print('Given string is not Binary!')
        break
else:
    print('Given string is Binary!')

Enter String: 123654

Given string is not Binary!
```

13) WAP to sort dictionary by key or value.

```
d1 = {102:'Shruti',101:'Riya',103:'Nandu'}
#sort by key
d2 = list(d1)
d2.sort()
print('Sorted by key:',d2)

#sort by value
d3 = list(d1.values())
d3.sort()
print('Sorted by value:',d3)
```

```
#sort by items(key:value)
d4 = list(d1.items())
d4.sort()
print('Sorted by item:',d4)
```

```
Sorted by key: [101, 102, 103]
Sorted by value: ['Nandu', 'Riya', 'Shruti']
Sorted by item: [(101, 'Riya'), (102, 'Shruti'), (103, 'Nandu')]
```

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

```
dict = {'a':100 ,'b':200 ,'c':300 ,'d':400 ,'e':500}
sum = 0
for i in dict:
    sum = sum + dict[i]
print('Sum of values od dictionary =',sum)
```

```
Sum of values od dictionary = 1500
```

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.

```
dict =  {'a':100 ,'b':200 ,'g':300 ,'d':400 ,'e':500}
key = input('Enter Key :')
if key in dict:
    print(dict[key])
else:
    print('Key Not Found!')
```

```
Enter Key : f

Key Not Found!
```

User Defined Function

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

```
mass = int(input('Enter mass:'))
height = int(input('Enter height:'))
def calBMI(mass,height):
    BMI = mass/height**2
    print('BMI =',BMI)

calBMI(mass,height)
```

```
Enter mass: 5
Enter height: 4

BMI = 0.3125
```

02) Write a function that add first n numbers.

```
n = int(input('Enter num :'))
def addNnumber(n):
    sum = 0
    for i in range(n+1):
        sum = sum + i
    print('Sum =',sum)

addNnumber(n)
```

```
Enter num : 5

Sum = 15
```

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

```
n = int(input('Enter number :'))
def primeOrNot(n):
    count = 0
    for i in range(2,(n//2)+1):
        if n%i==0:
            count+=1
    if count==0:
        return 1
    else:
        return 0

primeOrNot(n)
```

```
Enter number : 4
```

```
0
```

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

```
n = int(input('Enter strting number :'))
m = int(input('Enter ending number :'))
def primeInRange(n,m):
    for j in range(n,m+1):
        count = 0
        for i in range(2,(j//2)+1):
            if j%i==0:
                count+=1
                break
        if count==0:
            print(j)
primeInRange(n,m)
```

```
Enter strting number : 1
Enter ending number : 10
```

```
1
2
3
5
7
```

```
n = int(input('Enter strting number :'))
m = int(input('Enter ending number :'))
def primeOrNot(n):
    count = 0
    for i in range(2,(n//2)+1):
        if n%i==0:
            count+=1
    if count==0:
        return 1
    else:
        return 0
def Prime(n,m):
    for i in range(n,m+1):
        if primeOrNot(i)==1:
            print(i)

Prime(n,m)
```

```
Enter strting number : 1
Enter ending number : 10

1
2
3
5
7
```

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

```
str = input('Enter String :')
def palindromOrNot(str):
    if str == str[::-1]:
        return True
    else:
        return False

palindromOrNot(str)

Enter String : abcd

False
```

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

```
l1 = [1 ,2 ,3 ,4 ,5]
def sumOfList(l1):
    sum = 0
    for i in l1:
        sum = sum + i
    print('Sum =',sum)

sumOfList(l1)

Sum = 15
```

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

```
l1 = [(1,2,3),(2,3),(3,4),(4,5,6)]
def Sum(l1):
    sum = 0
    for i in l1:
        sum = sum + i[0]
    print('Sum =',sum)

Sum(l1)
```

```
Sum = 10
```

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

```python
# Input nth term
n = int(input("Enter the term number :"))
def fibonacci(n):
    if n <= 1:
        return n
    else:
        return fibonacci(n-1) + fibonacci(n-2)

print(f"The {n}th term of the Fibonacci series is : {fibonacci(n)}")
```

```
Enter the term number : 7

The 7th term of the Fibonacci series is : 13
```

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

```python
dict = {101:'Ajay', 102:'Rahul', 103:'Jay', 104:'Pooja'}
rno = int(input('Enter Rollno:'))
def getName(rno):
    if rno in dict:
        print(dict[rno])  #dict.values should be changed to dict[i]
    else:
        print('Roll no not found!')

getName(rno)
```

```
Enter Rollno: 105

Roll no not found!
```

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

```python
scores = [200, 456, 300, 100, 234, 678 ,600]
def sumOfScore(scores):
    sum = 0
    for i in scores:
        if i%10==0:
            sum = sum + i
    print('Sum =',sum)
```

```
sumOfScore(scores)

Sum = 1200
```

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'}

```
dict = {'a': 10, 'b':20, 'c':30, 'd':40}
def invertDict(dict):
    dict1 = {value:key for key,value in dict.items()}
    print(dict1)

invertDict(dict)

{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.

```
str1="the quick brown fox jumps over the lazy dog"
def pangram(str1):
    for i in 'qwertyuioplkjhgfdsazxcvbnm':
        if i not in str1:
            return False
    return True

pangram(str1)

True
```

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

```
s1 = 'AbcDEf@ghI'
def noOfUpperAndLower(s1):
    count1 = 0
    count2 = 0
    for i in s1:
        if i.isupper():
            count1+=1
        elif i.islower():
            count2+=1
    print('No of Uppercase letter in given string =',count1)
    print('No of Lowercase letter in given string =',count2)

noOfUpperAndLower(s1)

No of Uppercase letter in given string = 4
No of Lowercase letter in given string = 5
```

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

```
a = int(input('Enter num1 :'))
b = int(input('Enter num2 :'))
small = lambda a,b : a if a<b else b

print(small(a,b),'is smallest.')

Enter num1 : 10
Enter num2 : 5

5 is smallest.
```

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

```
l1 = ['Darshi','Shrutii','Riyariya','Nandaninandu','Diya','Mansi']
ans = list(filter(lambda x : len(x)>7,l1))
ans

['Riyariya', 'Nandaninandu']
```

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

```
l1 = ['darshi','shruti','riya','nandu','mansi']
ans = list(map(lambda name : name.capitalize(),l1))
ans

['Darshi', 'Shruti', 'Riya', 'Nandu', 'Mansi']
```

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

```python
# Positional Args
def positional_args(a, b):
    return a + b
result1 = positional_args(5, 3)
print("Addition By Positional Arguments:", result1)

#Keyword Args
def keyword_args(a, b):
    return a + b
result2 = keyword_args(a=5, b=3)
print("Addition By Keyword Arguments:", result2)

#Default Args
def default_args(a, b=10):
    return a + b
result3a = default_args(15)
result3b = default_args(15, 5)
print("Addition By Default Arguments:", result3a, result3b)

#Variable Legngth Positional(args) & variable length Keyword Arguments
(*kwargs)
def variable_args(*args, **kwargs):
    sum_args = sum(args)
    concatenated_kwargs = " ".join(f"{key}={value}" for key, value in
kwargs.items())
    return f"Sum of args: {sum_args}, Kwargs: {concatenated_kwargs}"
result4 = variable_args(1, 2, 3, name="Alice", age=25)
print("Variable Length Arguments:", result4)

#Keyword-Only & Positional Only Arguments
def mixed_args(a, /, b, *, c):
    return f"Positional-Only: {a}, Regular: {b}, Keyword-Only: {c}"
result5 = mixed_args(1, b=2, c=3)
print("Keyword-Only & Positional-Only Arguments:", result5)
```

```
Addition By Positional Arguments: 8
Addition By Keyword Arguments: 8
Addition By Default Arguments: 25 20
Variable Length Arguments: Sum of args: 6, Kwargs: name=Alice age=25
Keyword-Only & Positional-Only Arguments: Positional-Only: 1, Regular:
2, Keyword-Only: 3
```

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

```python
try:
    # Read as a single string
    with open("file1.txt", "r") as file:
        content = file.read()
        print("=== File Content as String ===")
        print(content)

    # Read line by line
    with open("file1.txt", "r") as file:
        print("\n=== File Content Line by Line ===")
        for line in file:
            print(line, end="")  # Avoids extra new lines

    # Read into a list
    with open("file1.txt", "r") as file:
        content_list = file.readlines()
        print("\n\n=== File Content as List ===")
        print(content_list)

except FileNotFoundError:
    print("Error: File not found. Check the path!")
```

```
=== File Content as String ===
10
20
50
20
10


=== File Content Line by Line ===
10
20
50
```

```
20
10


=== File Content as List ===
['10\n', '20\n', '50\n', '20\n', '10\n']
```

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

```
fp = open('new.txt','x')
fp.close()

#Error show because file is already existed

----------------------------------------------------------------------
-----
FileExistsError                           Traceback (most recent call
last)
Cell In[21], line 1
----> 1 fp = open('new.txt','x')
      2 fp.close()

File ~\anaconda3\Lib\site-packages\IPython\core\
interactiveshell.py:324, in _modified_open(file, *args, **kwargs)
    317 if file in {0, 1, 2}:
    318     raise ValueError(
    319         f"IPython won't let you open fd={file} by default "
    320         "as it is likely to crash IPython. If you know what
you are doing, "
    321         "you can use builtins' open."
    322     )
--> 324 return io_open(file, *args, **kwargs)

FileExistsError: [Errno 17] File exists: 'new.txt'
```

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

```
fp = open('new.txt','r')
for i in range(0,5):
    print(fp.readline())
fp.close()

Hi

Hello

How

Are
```

```
You?
```

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

```
fp = open("new.txt","r")
li = fp.readlines()
max = 0
for i in li:
    nw = i.split()
    for j in nw:
        if len(j)>max:
            max = len(j)
            lw = j
print(lw)
fp.close()

You?Is
```

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

```
fp = open("new.txt","r")

#number of lines
li = fp.readlines()
lines = len(li)-1
print('Number of lines =',lines)

#number of words
li2 = str(li).split()
words = len(li2)-1
print('Number of words =',words)

#number of characters
sum = 0
print(li)
for i in li:
    sum = sum + len(i)
print('Number of characters =',sum)

fp.close()

Number of lines = 5
Number of words = 6
['Hi\n', 'Hello\n', 'How\n', 'Are\n', 'You?\n', 'Kem Chho?\n']
Number of characters = 32
```

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

```
fp = open("new.txt","r")
fp1 = fp.read()
fp2 = open("file.txt","w")
fp2.write(fp1)
fp.close()
fp2.close()

print("File copied successfully!")
```

```
File copied successfully!
```

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

```
with open("file.txt", "rb") as fp:
    file_size = fp.seek(0, 2)
print(f"The size of the file is: {file_size} bytes")
fp.close()
```

```
The size of the file is: 40 bytes
```

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

```
fp = open("new.txt" , "r")
def NameFrequency(fp):
    nThis = 0
    nIs=0
    for i in fp:
        nThis += i.split().count('This')
        nIs += i.split().count('Is')
    print(nThis)
    print(nIs)
NameFrequency(fp)
fp.close()
```

```
1
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.

```
li = []
fp = open('file1.txt',"w+")
for i in range(0,5):
    score = int(input(f'Enter score for subject {i+1}: '))
    li.append(score)
    fp.write(str(score) + "\n")
fp.seek(0)  # Move file pointer to the beginning
print("Stored scores in file:")
print(fp.read())

fp.close()

max_score = max(li)
print('Highest score over these 5 subjects is :', max_score)
```

```
Enter score for subject 1:  10
Enter score for subject 2:  20
Enter score for subject 3:  50
Enter score for subject 4:  20
Enter score for subject 5:  10

Stored scores in file:
10
20
50
20
10

Highest score over these 5 subjects is : 50
```

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

(Note: each number should be in new line)

```
fp = open("primenumbers.txt","w+")
num = 1
count_p = 0
while count_p<100:
    count = 0
    for i in range(2,(num//2)+1):
        if num%i==0:
            count+=1
    if count==0:
        fp.write(str(num) + "\n")
        count_p+=1
    num+=1
fp.close()
print("First 100 prime numbers written to primenumbers.txt
successfully.")
```

```
First 100 prime numbers written to primenumbers.txt successfully.
```

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

```
fp1 = open("file.txt","r")
content1 = fp1.read()

fp2 = open("file1.txt","r")
content2 = fp2.read()

final_content = content1 + "\n" + content2

fp = open("file2.txt","w")
fp.write(final_content)
fp.close()
print('File merged successfully.')
```

```
File merged successfully.
```

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

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

```
fp = open("sampletellseek.txt", "w")
fp.write("Hello, this is a sample text file.")
fp.close()

fp = open("sampletellseek.txt", "rb")

print("Initial position:", fp.tell())  # Position at the beginning (0)

# Seek from the beginning (use 0)
fp.seek(7, 0)
print("After seeking 7 bytes from the beginning:", fp.tell())

# Seek from the current position (use 1)
fp.seek(5, 1)
print("After seeking 5 bytes forward from current position:",
fp.tell())

# Seek from the end (use 2)
fp.seek(-10, 2)
print("After seeking 10 bytes back from the end:", fp.tell())

# Reading from the final position
print("Remaining text from current position:", fp.read().decode())
```

```
fp.close()
```

```
Initial position: 0
After seeking 7 bytes from the beginning: 7
After seeking 5 bytes forward from current position: 12
After seeking 10 bytes back from the end: 24
Remaining text from current position: text file.
```

python-programming-lab-10

March 11, 2025

Python Programming - 2301CS404

Lab - 10

Shruti | 23010101311 | 03-02-2025

1 Exception Handling

1.0.1 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.

[6]:
```
try:
    a = int(input('Enter a :'))
    b = int(input('Enter b :'))
    ans = a//b
    print('Answer =',ans)
    print(a + 'b')
except ZeroDivisionError:
    print('ZeroDivisionError Occured!')
except ValueError:
    print('ValueError Occured!')
except TypeError:
    print('TypeError Occured!')
```

```
Enter a : 10
Enter b : a

ValueError Occured!
```

[27]:
```
try:
    a = int(input('Enter a :'))
    b = int(input('Enter b :'))
    ans = a//b
    print('Answer =',ans)
    print(a + 'b')
except (ZeroDivisionError,ValueError,TypeError) as Error:
    print(Error)
```

```
Enter a : 10
Enter b : 2

Answer = 5
unsupported operand type(s) for +: 'int' and 'str'
```

1.0.2 02) WAP to handle following exceptions:

1. IndexError
2. KeyError

[37]:
```
li = [1 ,2 ,3 ,4 ,5]
try:
    print('li[4] =',li[4])
    print(li[7])
except IndexError:
    print('IndexError Occured!')
```

```
li[4] = 5
IndexError Occured!
```

[51]:
```
d1 = {1:'a' ,2:'b' ,3:'c'}
try:
    print('d1[3] =',d1[3])
    print(d1[5])
except KeyError as err:
    print('KeyError Occured! :',err)#When keyerror occurs it will print the key␣
  ↪passed in input
```

```
d1[3] = c
KeyError Occured! : 5
```

1.0.3 03) WAP to handle following exceptions:

1. FileNotFoundError
2. ModuleNotFoundError

[57]:
```
try:
    fp = open("abcd.txt","r")
    print(fp.read())
except FileNotFoundError as msg:
    print('FileNotFoundError :',msg)
```

```
FileNotFoundError : [Errno 2] No such file or directory: 'abcd.txt'
```

[60]:
```
try:
    import maths
```

```
except ModuleNotFoundError as msg:
    print('ModuleNotFoundError :',msg)
```

```
ModuleNotFoundError : No module named 'maths'
```

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

[86]:
```
try:
    a = int(input('Enter a :'))
    b = input('Enter b :')
    #ans = a//b
    #print('Answer =',ans)
    print(a + 'b')
    #print(b[5])
    print(c)
except (ZeroDivisionError,TypeError,ValueError,NameError,IndexError) as err:
    print(err)
```

```
Enter a : 10
Enter b : asdfg
```

```
unsupported operand type(s) for +: 'int' and 'str'
```

1.0.5 05) WAP to demonstrate else and finally block.

[92]:
```
try:
    fp = open('abc.txt','r')
except FileNotFoundError as err:
    print(err)
else:
    print(fp.read())
    fp.close()
finally:
    print('This block is always executes!')
```

```
hello
This block is always executes!
```

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

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

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

[142]:
```
try:
    grade = input('Enter string with comma separated :')
    li = [int(i) for i in grade.split(',')]
    print(li)
except ValueError as msg:
    print('ValueError :',msg)
```

```
Enter string with comma separated : a,b,c,d
```

```
ValueError : invalid literal for int() with base 10: 'a'
```

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

[98]:
```
def divide(a,b):
    try:
        print('Division :',a//b)
    except ZeroDivisionError as err:
        print('ZeroDivisionError :',err)

divide(10,2)
divide(10,0)
```

```
Division : 5
ZeroDivisionError : integer division or modulo by zero
```

1.0.10 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.

[118]:
```
try:
    age = int(input('Enter Age :'))
    if age<18:
        #print('Enter Valid Age!')
        raise ValueError ('Enter Valid Age!')
    else:
        print(age)
except ValueError as err:
    print(err)
```

```
Enter Age : 17

Enter Valid Age!
```

1.0.11 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.

[128]:
```python
class InvalidUsernameError(Exception):
    pass
try:
    Name = input('Enter Username :')
    if (len(Name)-1)<5 or (len(Name)-1)>15:
        raise InvalidUsernameError
    else:
        print('Usename =',Name)
except InvalidUsernameError:
    print('Username must be between 5 and 15 characters long!')
```

```
Enter Username : qwertyuioplkjhgfdsa

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

1.0.12 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.

[136]:
```python
import math
class NegativeNumberError(Exception):
    def __init__(self,msg):
        self.msg = msg
try:
    a = int(input('Enter num :'))
    if a<0:
        raise NegativeNumberError('Cannot calculate the square root of a␣
 ↪negative number!')
    else:
        print('Sqrt of',a,'=',math.sqrt(a))
except NegativeNumberError as error:
    print(error)
```

```
Enter num : -4

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

python-programming-lab-11

March 11, 2025

Python Programming - 2301CS404

Lab - 11

Shruti | 23010101311 | 10-02-2025

1 Modules

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

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

[54]:
```
import calculator
a = float(input("Enter first number: "))
b = float(input("Enter second number: "))

print('Addition =',calculator.Addition(a,b))
print('Subtraction =',calculator.Subtraction(a,b))
print('Multiplictaion =',calculator.Multiplication(a,b))
print('Division =',calculator.Division(a,b))
```

```
Enter first number:  10
Enter second number:  2
```

```
Addition = 12.0
Subtraction = 8.0
Multiplictaion = 20.0
Division = 5.0
```

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

[20]:
```
import random
str = input('Enter String :')
print('Random character from the given string :',random.choice(str))
```

```
Enter String : Shruti
Random character from the given string : r
```

[24]:
```
import random
str = input('Enter String :')
n = random.randrange(0,len(str))
print('Random character from the given string :',str[n])
```

```
Enter String : shruti
Random character from the given string : h
```

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

[22]:
```
li = [1 ,2 ,3 ,4 ,5]
print('Random element from the given list :',random.choice(li))
```

```
Random element from the given list : 5
```

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

[63]:
```
random.seed(5)
print('Random same number :',random.randint(1,6))
```

```
Random same number : 5
```

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

[84]:
```
print('Random number between 100 and 999 which is divisible by 5 is =')
for i in range(0,3):
    n = random.randrange(100,999,5)
    print(n)
```

```
Random number between 100 and 999 which is divisible by 5 is =
575
700
415
```

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

[101]:
```
li = []
for i in range(0,100):
    n = random.randint(1000,9999)
    li.append(n)

winner = random.choice(li)
runner_up = random.choice(li)

while winner == runner_up:
    runner_up = random.choice(li)
```

```
print('Runner up is whose lottery ticket number is',runner_up)
print('Winner is whose lottery ticket number is',winner)
```

```
Runner up is whose lottery ticket number is 9870
Winner is whose lottery ticket number is 9030
```

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

[103]:
```
import datetime
print('Current date and time is :',datetime.datetime.today())
```

```
Current date and time is : 2025-02-10 13:19:37.756838
```

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

[ ]:
```
date = input("Enter a date :")
convert_date = datetime.datetime.strptime(date,"%d/%m/%Y")
subtract_date = convert_date - datetime.timedelta(days=7)
print("Subtract a week",subtract_date)
```

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

[58]:
```
import datetime

date1 = input('Enter first date (DD-MM-YYYY): ')
date2 = input('Enter second date (DD-MM-YYYY): ')

d1 = datetime.datetime.strptime(date1, '%d-%m-%Y')
d2 = datetime.datetime.strptime(date2, '%d-%m-%Y')

difference = abs((d2 - d1).days)

print('Number of days between given dates is:', difference)
```

```
Enter first date (DD-MM-YYYY):  12-02-2024
Enter second date (DD-MM-YYYY):  15-02-2025

Number of days between given dates is: 369
```

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

[12]:
```
date=input("Enter a date1: ")
convert_date = datetime.datetime.strptime(date,"%d/%m/%Y")

print(convert_date)
```

```
d1=convert_date.strftime("%A")
print("day of the week of a given date :",d1)
```

```
Enter a date1:  2/02/2025

2025-02-02 00:00:00
day of the week of a given date : Sunday
```

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

```
[1]: import datetime
     print(datetime.datetime.today())
```

```
2025-02-16 18:55:23.250809
```

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

```
[8]: import math
     print('Vlue Of Pi =',math.pi)
     print('Floor =',math.floor(22.22))
     print('Ceil =',math.ceil(22.22))
```

```
Vlue Of Pi = 3.141592653589793
Floor = 22
Ceil = 23
```

```
[ ]:
```

python-programming-lab-12

March 11, 2025

Python Programming - 2301CS404

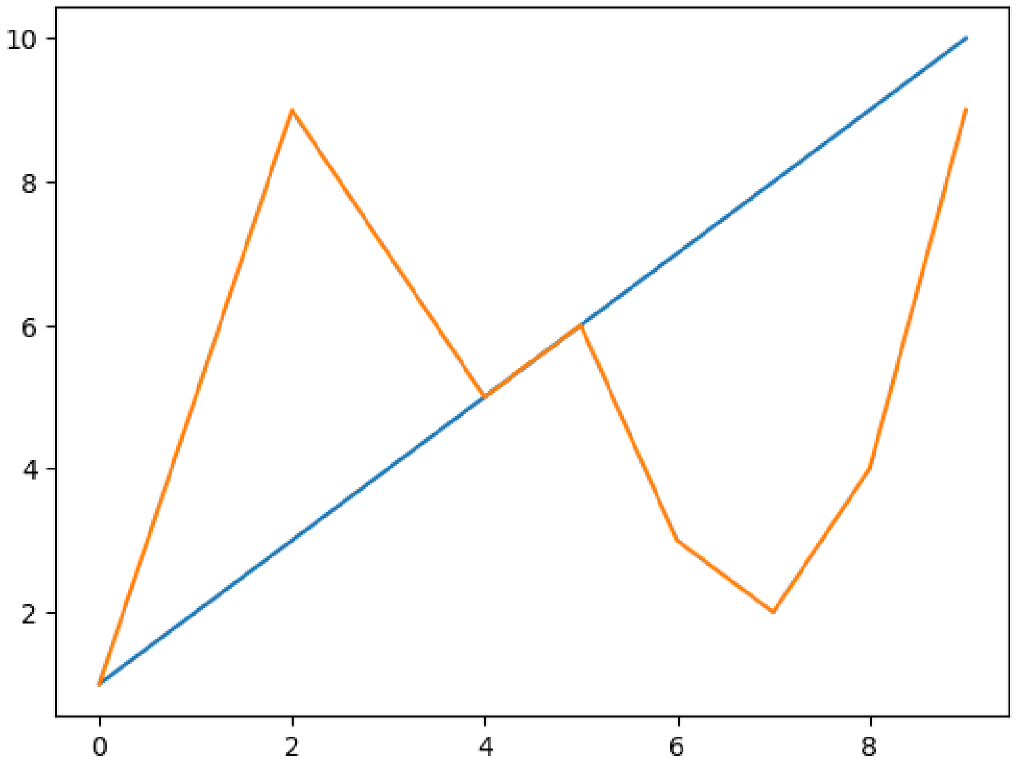
Lab - 12

Shruti | 23010101311 | 17-02-2025

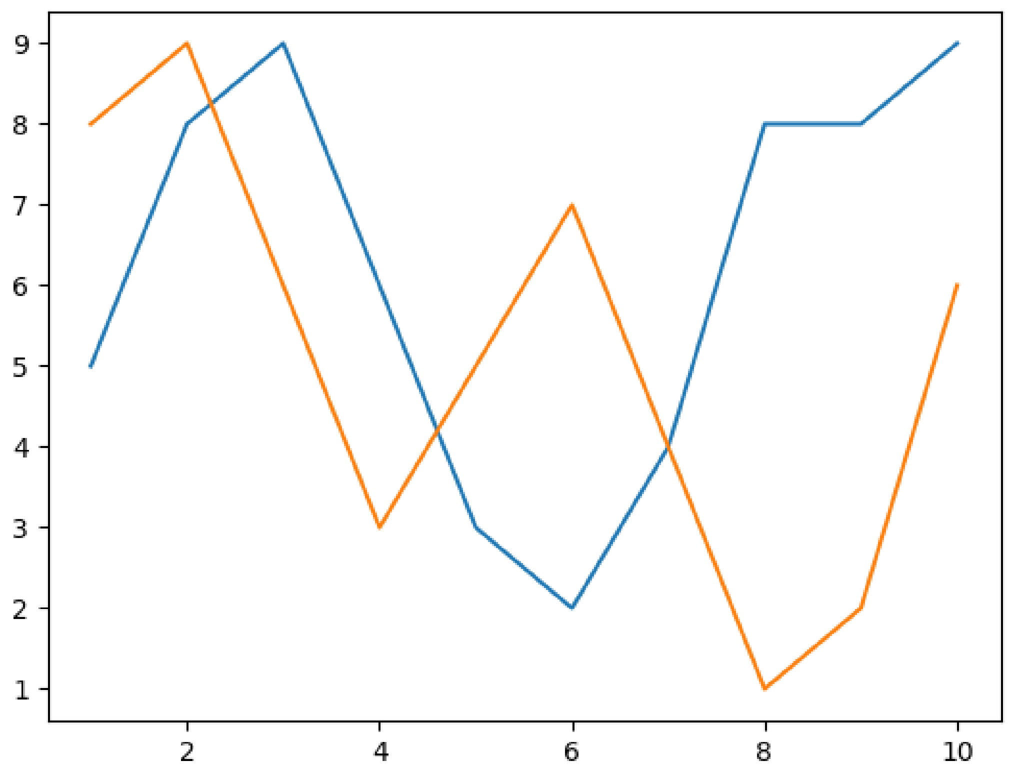
```
[ ]: #import matplotlib below
```

```
[6]: import matplotlib.pyplot as plt
```

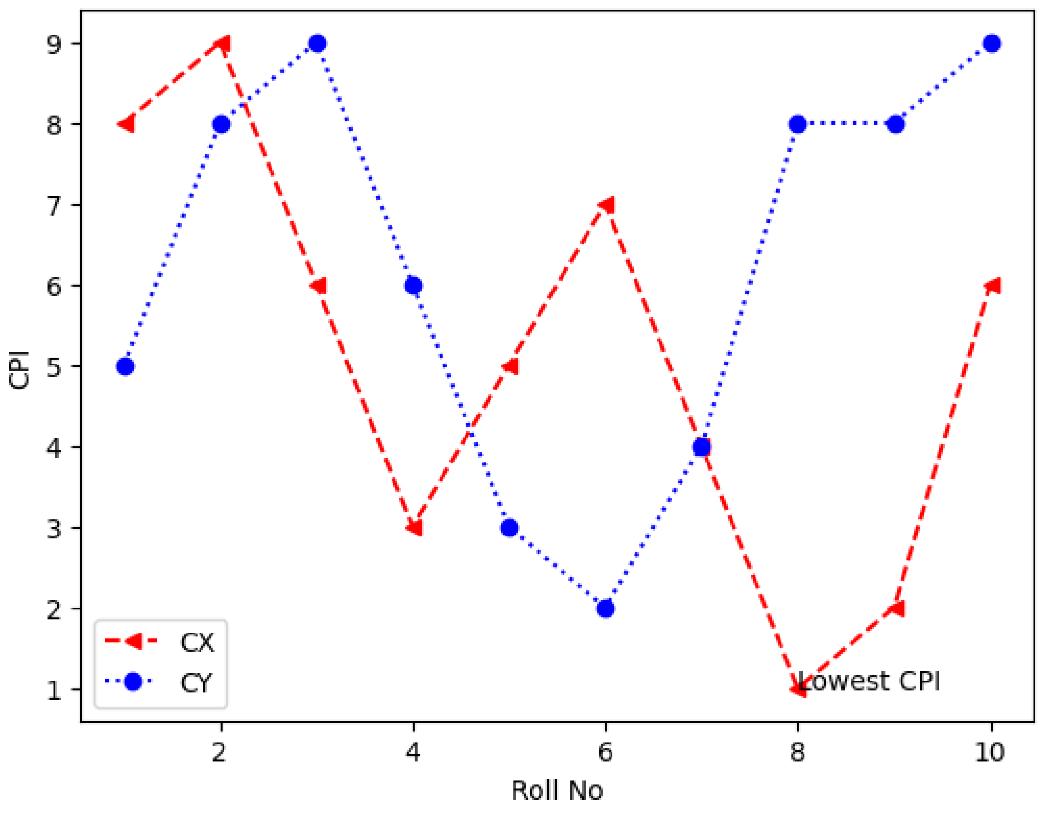
```
[10]: x = range(1,11)
      y = [1,5,9,7,5,6,3,2,4,9]

      # write a code to display the line chart of above x & y
      plt.plot(x)
      plt.plot(y)
      plt.show()
```



[12]:
```
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)
plt.plot(x,cyMarks)
plt.show()
```

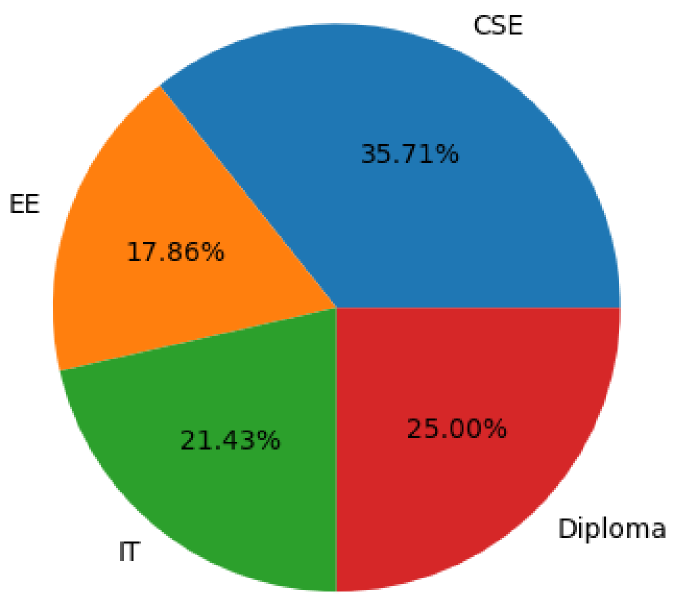


```
[44]: x = range(1,11,1)
      cxMarks= [8,9,6,3,5,7,4,1,2,6]
      cyMarks= [5,8,9,6,3,2,4,8,8,9]



      # write a code to generate below graph

      plt.plot(x,cxMarks,linestyle='--',color="red",marker='<')
      plt.plot(x,cyMarks,linestyle=':',color='blue',marker='o')
      plt.xlabel('Roll No')
      plt.ylabel('CPI')
      plt.legend(['CX','CY'])
      plt.annotate('Lowest CPI',xy=[8,1])
      plt.show()
```



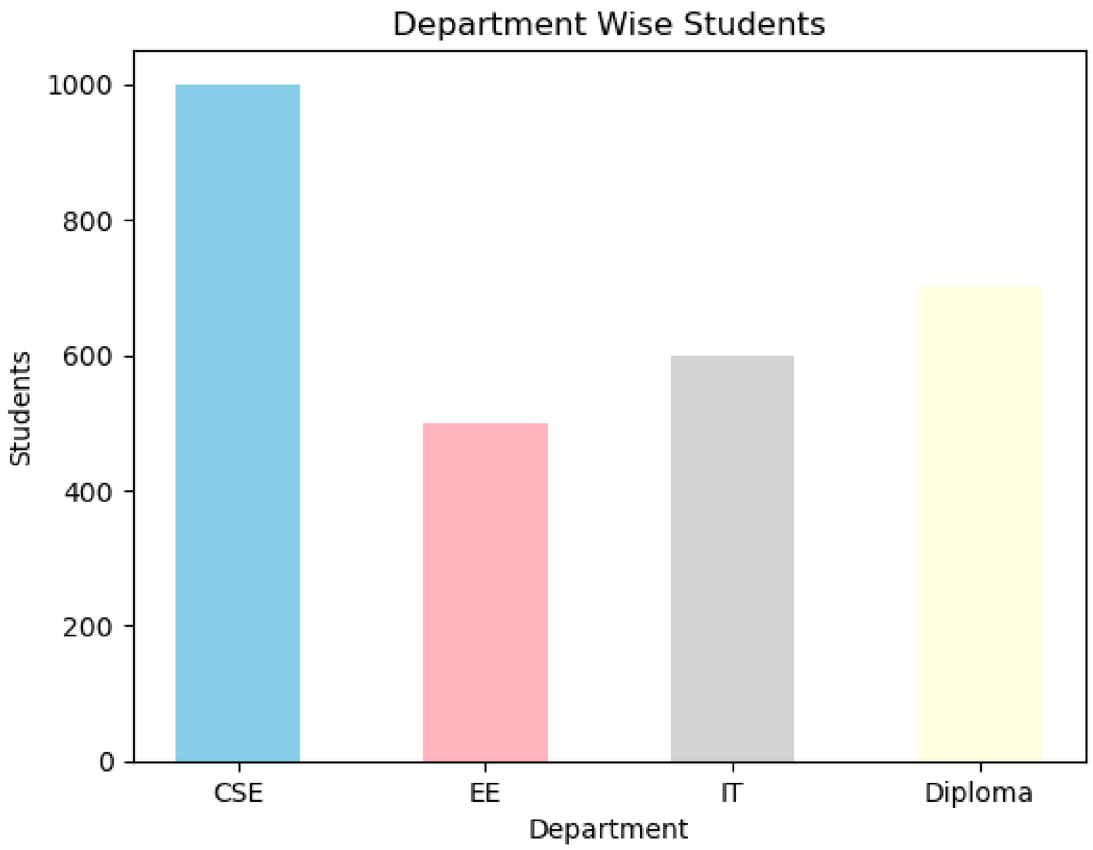
0.0.1 04) WAP to demonstrate the use of Pie chart.

[88]:
```
dept = ['CSE','EE','IT','Diploma']
stu = [1000, 500, 600, 700]
plt.pie(stu,labels=dept,autopct='%1.2f%%')
plt.show()
```



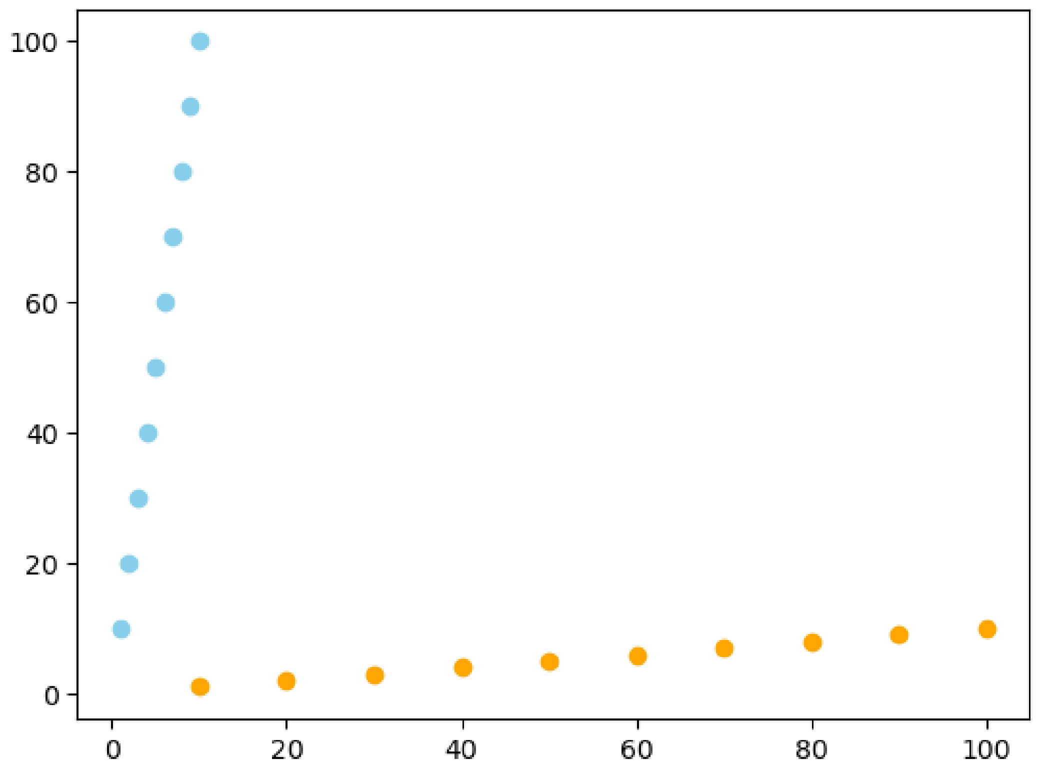
0.0.2 05) WAP to demonstrate the use of Bar chart.

[118]:
```
dept = ['CSE','EE','IT','Diploma']
stu = [1000, 500, 600, 700]
clr = ['skyblue','lightpink','lightgray','lightyellow']
plt.bar(dept,stu,width=0.5,color=clr)
plt.xlabel('Department')
plt.ylabel('Students')
plt.title('Department Wise Students')
plt.show()
```



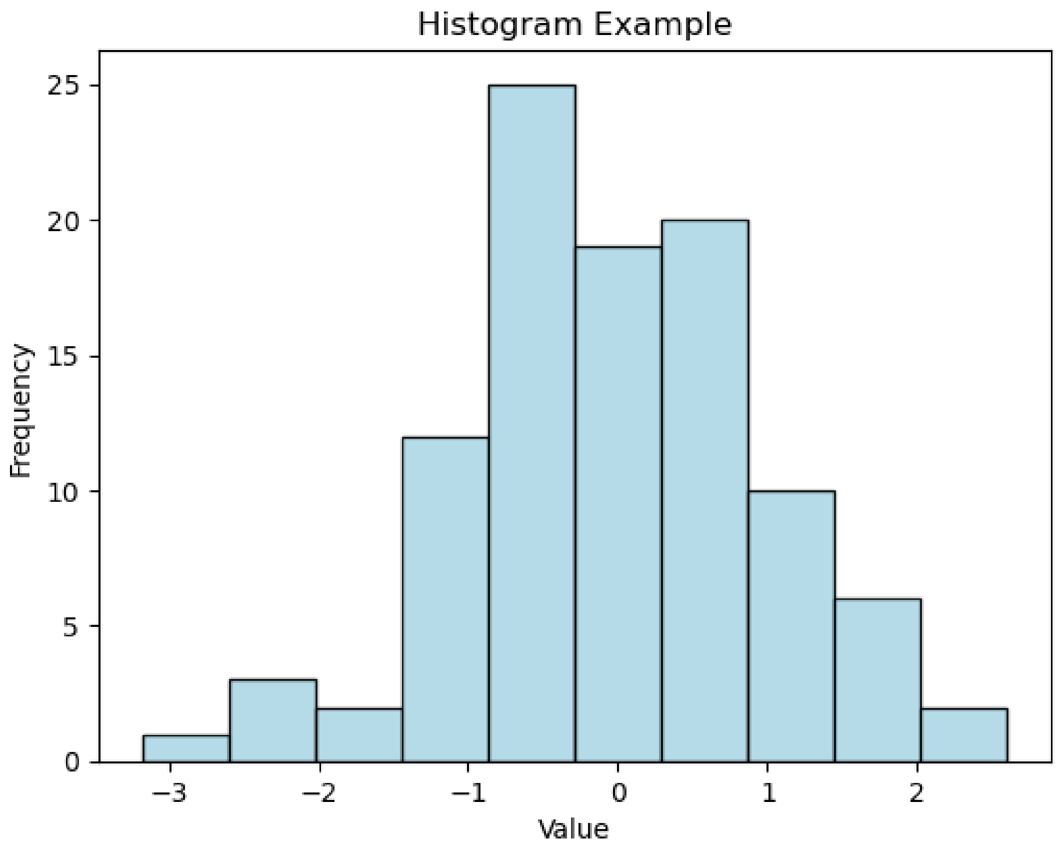
0.0.3 06) WAP to demonstrate the use of Scatter Plot.

```
[138]: x = [1,2,3,4,5,6,7,8,9,10]
       y = [10,20,30,40,50,60,70,80,90,100]
       plt.scatter(x,y,color='skyblue')
       plt.scatter(y,x,color='orange')
       plt.show()
```



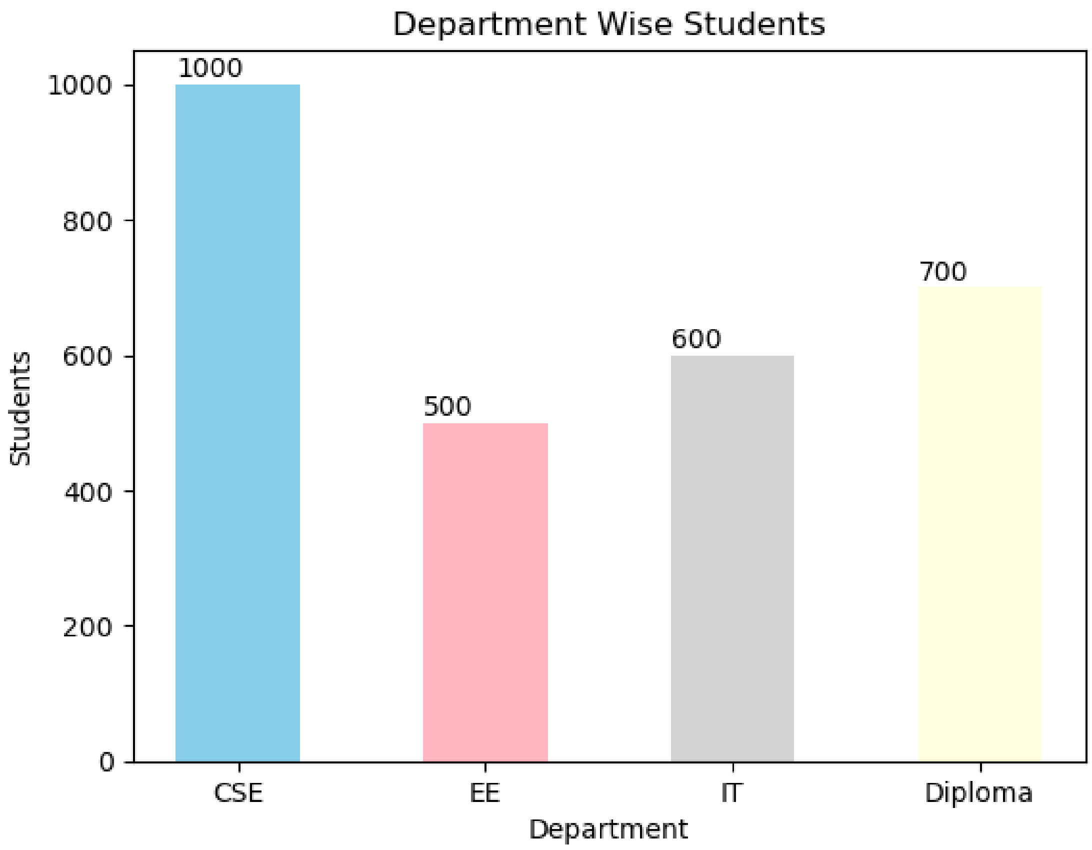
0.0.4 07) WAP to demonstrate the use of Histogram.

[178]:
```python
import matplotlib.pyplot as plt
import numpy as np

data = np.random.randn(100)
plt.hist(data, bins=10, color='lightblue', edgecolor='black', alpha=0.9)

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



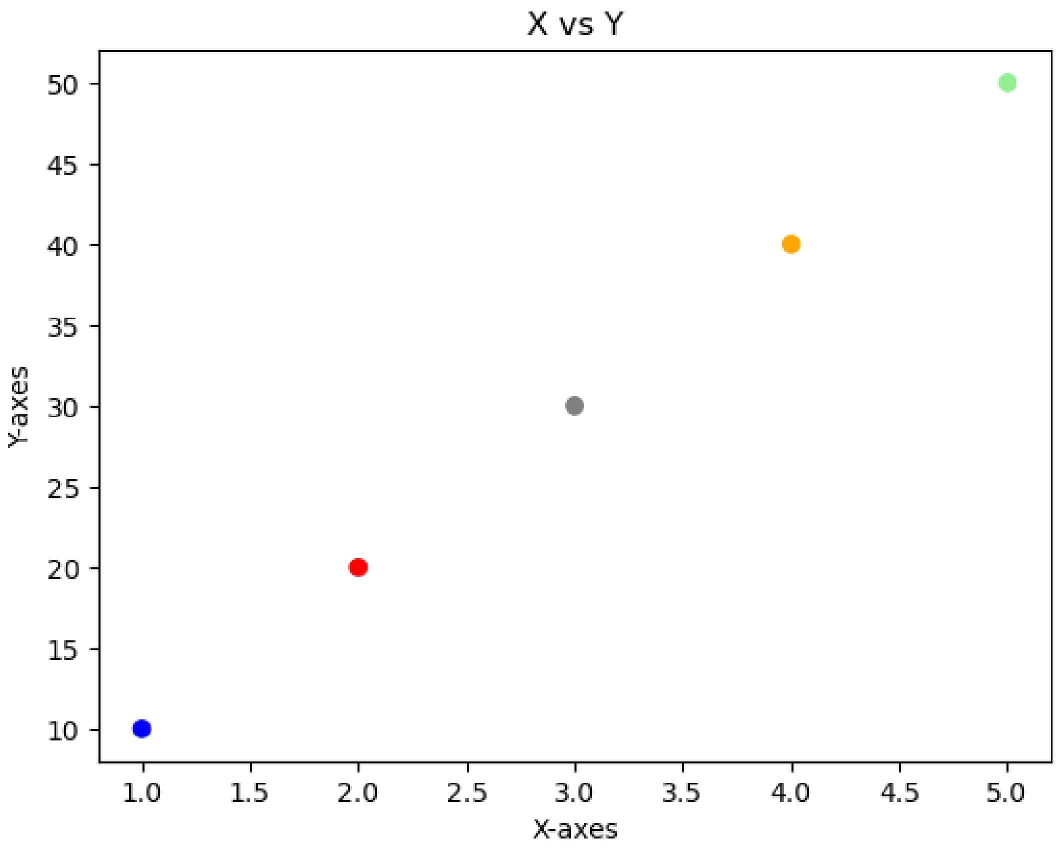
0.0.5 08) WAP to display the value of each bar in a bar chart using Matplotlib.

[146]:
```python
dept = ['CSE','EE','IT','Diploma']
stu = [1000, 500, 600, 700]
clr = ['skyblue','lightpink','lightgray','lightyellow']
bar = plt.bar(dept,stu,width=0.5,color=clr)
plt.xlabel('Department')
plt.ylabel('Students')
plt.title('Department Wise Students')

for i in bar:
    yc = i.get_height()
    plt.text(i.get_x(),yc+10,f"{yc}")

plt.show()
```



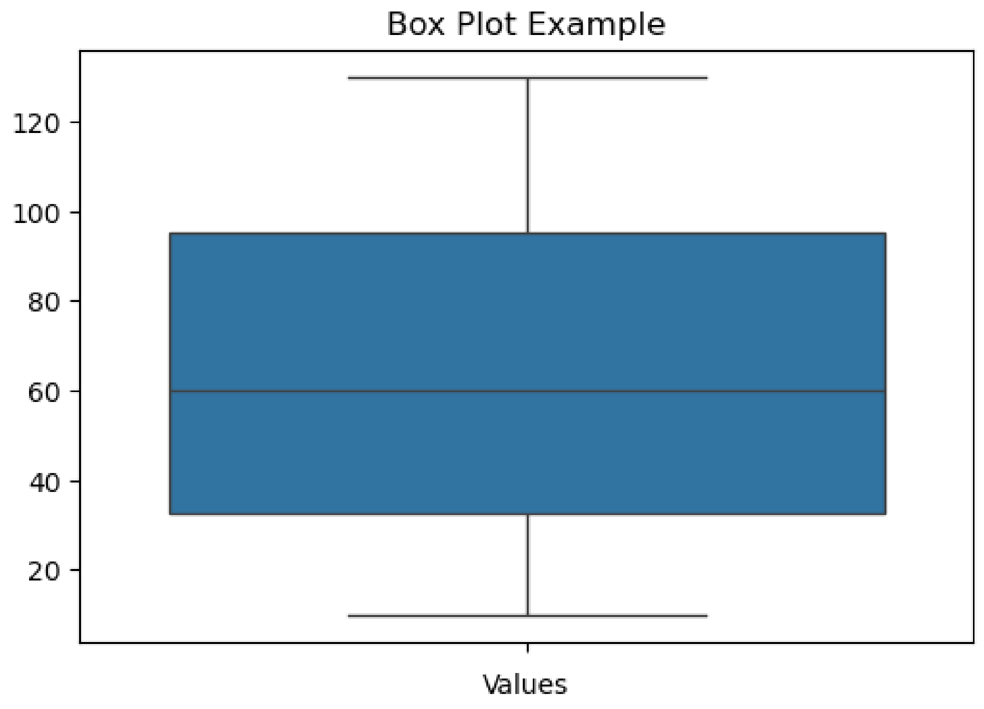
0.0.6 09) WAP create a Scatter Plot with several colors in Matplotlib?

[158]:
```
x = [1,2,3,4,5]
y = [10 ,20 ,30 ,40 ,50]
clr = ['blue','red','gray','orange','lightgreen']
plt.scatter(x,y,color=clr)
plt.xlabel('X-axes')
plt.ylabel('Y-axes')
plt.title('X vs Y')
plt.show()
```



0.0.7 10) WAP to create a Box Plot.

[160]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Sample Data
data = [10, 20, 25, 30, 35, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130]

# Creating the Box Plot
plt.figure(figsize=(6,4))
sns.boxplot(data=data)

# Title and Labels
plt.title("Box Plot Example")
plt.xlabel("Values")

# Show the plot
plt.show()
```


Box Plot Example
120
100
80
60
40
20
Values

```
[ ]:
```

python-programming-lab-13-1

March 11, 2025

Python Programming - 2301CS404

Lab - 13

Shruti | 23010101311 | 03-03-2025

1 OOP

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

[9]:
```
class Students:
    def __init__(self,name,age,grade):
        self.Name = name
        self.Age = age
        self.Grade = grade

s1 = Students('Shruti',19,'A++')
print("Name :",s1.Name)
print("Age :",s1.Age)
print("Grade :",s1.Grade)
```

```
Name : Shruti
Age : 19
Grade : A++
```

1.0.2 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.

[13]:
```
class Bank_Account:
    Account_No = ''
    User_Name = ''
    Email = ''
    Account_Type = ''
    Account_Balance = 0
    def␣
↪GetAccountDetails(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 DisplayAccountDetails(self):
        print("Account_No :",self.Account_No)
        print("User_Name :",self.User_Name)
        print("Email :",self.Email)
        print("Account_Type :",self.Account_Type)
        print("Account_Balance :",self.Account_Balance)

#This is for main method
if __name__ == '__main__':
    Bn = Bank_Account()
    Bn.GetAccountDetails('123654987','Shruti','shruti@gmail.
↪com','valid',20000000)
    Bn.DisplayAccountDetails()
```

```
Account_No : 123654987
User_Name : Shruti
Email : shruti@gmail.com
Account_Type : valid
Account_Balance : 20000000
```

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

[32]:
```
import math
class Circle:
    def __init__(self,r):
        self.Radius = r

    def Area(self):
        area = math.pi*self.Radius*self.Radius
        print("Area of circle =",area)

    def Perimeter(self):
        peri = 2*math.pi*self.Radius
        print("Perimeter of circle =",peri)

c = Circle(12)
c.Area()
```

```
c.Perimeter()
```

```
Area of circle = 452.3893421169302
Perimeter of circle = 75.39822368615503
```

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

[1]:
```
class Employee:
    def __init__(self, name, age, salary):
        self.name = name
        self.age = age
        self.salary = salary

    def update_info(self, name=None, age=None, salary=None):
        if name:
            self.name = name
        if age:
            self.age = age
        if salary:
            self.salary = salary

    def display_info(self):
        return f"Name: {self.name}, Age: {self.age}, Salary: {self.salary}"

# Example usage
emp1 = Employee("Alice", 30, 50000)
print(emp1.display_info())  # Output: Name: Alice, Age: 30, Salary: $50000

print("\nupdate information......\n")
emp1.update_info(age=31, salary=55000)
print(emp1.display_info())  # Output: Name: Alice, Age: 31, Salary: $55000
```

```
Name: Alice, Age: 30, Salary: 50000

update information…

Name: Alice, Age: 31, Salary: 55000
```

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

[58]:
```
class Account:
    Balance = 0

    def Deposit(self,Balance,Money):
        self.Balance = Balance
        Balance += Money
        print("After Deposit Balance =",Balance)

    def Withdraw(self,Balance,Money):
        self.Balance = Balance
        Balance -= Money
        print("After Withdraw Balance =",Balance)

    def CheckBalance(self):
        print("Account Balance =",self.Balance)

a = Account()
a.Deposit(20000,1200)
a.Withdraw(20000,1200)
a.CheckBalance()
```

```
After Deposit Balance = 21200
After Withdraw Balance = 18800
Account Balance = 20000
```

[114]:
```
class Account:
    def __init__ (self,current_balance,money):
        self.current_balance = current_balance
        self.money = money

    def Deposit(self):
        self.current_balance += self.money
        print("After Deposit Balance =",self.current_balance)

    def Withdraw(self):
        if (self.current_balance>=self.money):
            self.current_balance -= self.money
        print("After Withdraw Balance =",self.current_balance)

    def CheckBalance(self):
        print("Account Balance =",self.current_balance)

a = Account(20000,1200)
a.Deposit()
a.Withdraw()
a.CheckBalance()
```

```
After Deposit Balance = 21200
After Withdraw Balance = 20000
Account Balance = 20000
```

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

[6]:
```python
class Managing_inventory:
    def _init_ (self):
        pass

    def AddItems(self):
        self.name = input("Enter your items name: ")
        self.price = float(input("Enter item price :"))
        self.quantity = int(input("Enter a quantity of your item:"))

    def DisplayItem(self):
        print("Item name is :",self.name)
        print("Item price is :",self.price)
        print("Item quantity is : ",self.quantity)

    def  UpdateItem(self,name=None,price=None,quantity=None):
        if name:
            self.name = name
        if price:
            self.price = price
        if quantity:
            self.quantity = quantity

    def DeleteItem(self):
        self.name = None
        self.price = None
        self.quantity = None

items = Managing_inventory()
items.AddItems()

print("\n after add data...\n")
items.DisplayItem()

print("\n after update Items........\n")
items.UpdateItem(name='laptop' , price =1000000)
items.DisplayItem()

print("\n after delete Items........\n")
items.DeleteItem()
items.DisplayItem()
```

```
Enter your items name:  Iphone
Enter item price : 2000000
Enter a quantity of your item: 2
```

```
 after add data…

Item name is : Iphone
Item price is : 2000000.0
Item quantity is :  2

 after update Items…

Item name is : laptop
Item price is : 1000000
Item quantity is :  2

 after delete Items…

Item name is : None
Item price is : None
Item quantity is :  None
```

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

[62]:
```
class Students:
    def __init__ (self,name,age,spi):
        self.name = name
        self.age = age
        self.spi = spi

s1 = Students('Shruti',18,'9.42')
print('Name :',s1.name)
print('Age :',s1.age)
print('SPI :',s1.spi)
```

```
Name : Shruti
Age : 18
SPI : 9.42
```

1.0.8 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.

[3]:
```python
class Student_kit:
    principal_name = "Mr. ABC"

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

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

    def generate_certificate(self):
        return (f"Certificate of Attendance\n"
                f"This is to certify that {self.name} has attended {self.
 ↪attendance_days} days of class.\n"
                f"Principal: {self.principal_name}")

# Example usage
student1 = Student_kit("John Doe")
student1.attendance(25)
print(student1.generate_certificate())
```

```
Certificate of Attendance
This is to certify that John Doe has attended 25 days of class.
Principal: Mr. ABC
```

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

[106]:
```python
class Time:
    def __init__(self, hour, minute):
        self.hour = hour
        self.minute = minute

    def add(self, other):
        total_minutes = self.minute + other.minute
        extra_hours = total_minutes // 60
        minutes = total_minutes % 60
        hours = self.hour + other.hour + extra_hours
        return Time(hours, minutes)

    def display(self):
        print(f"{self.hour} hours {self.minute} minutes")

time1 = Time(2, 50)
time2 = Time(1, 30)
time3 = time1.add(time2)
time3.display()
```

```
4 hours 20 minutes
```

```
[ ]:
```

python-programming-lab-13-2

March 11, 2025

Python Programming - 2301CS404

Lab - 13

Shruti | 23010101311 | 10-03-2025

0.1 Continued..

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

[1]:
```python
class Rectangle:
    def __init__(self, length, width):
        self.length = length
        self.width = width

    def area(self):
        return self.length * self.width

def calculate_area(rect):
    return rect.area()

rect1 = Rectangle(10, 15)

print("Area of the rectangle:", calculate_area(rect1))
```

```
Area of the rectangle: 150
```

0.1.2 11) Calculate the area of a square.

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

[5]:
```python
class Square:
    def __init__(self, side):
        self.side = side

    def area(self):
        a = self.side * self.side
        self.output(a)
    def output(self, a):
        print(f"Area of the square: {a}")

n = int(input('Enter n:'))
sq = Square(n)
sq.area()
```

```
Enter n: 15
```

```
Area of the square: 225
```

0.1.4 12) Calculate the area of a rectangle.

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

0.1.6 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.

[15]:
```
class Rectangle:
    def __init__(self, length, width):
        if length == width:
            print("THIS IS SQUARE.")
        else:
            self.length = length
            self.width = width

    def area(self):
        return self.output(self.length * self.width)

    def output(self, area):
        print("Area of the square: ",area)

    @classmethod
    def validate(cls, length, width):
        if length == width:
            print("THIS IS SQUARE.")
            return None
        return cls(length, width)

side1 = int(input("Enter side1:"))
side2 = int(input("Enter side2:"))
rect1 = Rectangle.validate(side1, side2)
if rect1:
    rect1.area()

side1 = int(input("Enter side1:"))
side2 = int(input("Enter side2:"))
```

```
rect2 = Rectangle.validate(side1, side2)
if rect2:
    rect2.area()
```

```
Enter side1: 12
Enter side2: 6

Area of the square:  72

Enter side1: 12
Enter side2: 12

THIS IS SQUARE.
```

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

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

[19]:
```
class Square:
    def __init__(self, side):
        self.__side = side  # Private attribute

    def get_side(self):
        return self.__side

    def set_side(self, newside):
        if newside > 0:
            self.__side = newside
        else:
            print("Side length must be positive!")

sq = Square(50)

print("Initial Side:", sq.get_side())

sq.set_side(80)
print("Updated Side:", sq.get_side())

sq.set_side(-3)  # This will display an error message
```

```
Initial Side: 50
Updated Side: 80
Side length must be positive!
```

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

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

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

[25]:
```python
class Profit:
    def getProfit(self):
        self.profit = float(input("Enter profit amount: "))

class Loss:
    def getLoss(self):
        self.loss = float(input("Enter loss amount: "))

class BalanceSheet(Profit, Loss):
    def __init__(self):
        self.profit = 0
        self.loss = 0
        self.balance = 0

    def getBalance(self):
        self.balance = self.profit - self.loss

    def printBalance(self):
        print("Net Balance:",self.balance)

bs = BalanceSheet()

bs.getProfit()
bs.getLoss()

bs.getBalance()
bs.printBalance()
```

```
Enter profit amount:  1500
Enter loss amount:  150

Net Balance: 1350.0
```

0.1.12 15) WAP to demonstrate all types of inheritance.

[27]:
```python
# 1. Single Inheritance
class Parent:
    def show(self):
        print("Single Inheritance: Parent class")
```

```python
class Child(Parent):
    pass

# 2. Multiple Inheritance
class Father:
    def fatherFeature(self):
        print("Multiple Inheritance: Feature from Father")

class Mother:
    def motherFeature(self):
        print("Multiple Inheritance: Feature from Mother")

class Child2(Father, Mother):
    pass

# 3. Multilevel Inheritance
class Grandparent:
    def grandFeature(self):
        print("Multilevel Inheritance: Feature from Grandparent")

class Parent2(Grandparent):
    pass

class Child3(Parent2):
    pass

# 4. Hierarchical Inheritance
class Base:
    def baseFeature(self):
        print("Hierarchical Inheritance: Base class feature")

class Derived1(Base):
    pass

class Derived2(Base):
    pass

# 5. Hybrid Inheritance (Combination of multiple types)
class A:
    def featureA(self):
        print("Hybrid Inheritance: Feature A")

class B(A):
    def featureB(self):
        print("Hybrid Inheritance: Feature B")

class C(A):
    def featureC(self):
        print("Hybrid Inheritance: Feature C")

class D(B, C):
    pass

print("\n--- Single Inheritance ---")
c1 = Child()
c1.show()

print("\n--- Multiple Inheritance ---")
c2 = Child2()
c2.fatherFeature()
c2.motherFeature()

print("\n--- Multilevel Inheritance ---")
c3 = Child3()
c3.grandFeature()

print("\n--- Hierarchical Inheritance ---")
d1 = Derived1()
d2 = Derived2()
d1.baseFeature()
d2.baseFeature()

print("\n--- Hybrid Inheritance ---")
d = D()
d.featureA()
d.featureB()
d.featureC()
```

```
--- Single Inheritance ---
Single Inheritance: Parent class

--- Multiple Inheritance ---
Multiple Inheritance: Feature from Father
Multiple Inheritance: Feature from Mother

--- Multilevel Inheritance ---
Multilevel Inheritance: Feature from Grandparent

--- Hierarchical Inheritance ---
Hierarchical Inheritance: Base class feature
Hierarchical Inheritance: Base class feature

--- Hybrid Inheritance ---
```

```
Hybrid Inheritance: Feature A
Hybrid Inheritance: Feature B
Hybrid Inheritance: Feature C
```

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

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

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

[29]:
```python
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

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

    def display(self):
        print("Name:",self.name, "\nAge:", self.age, "\nSalary:", self.salary)

emp = Employee("Shruti", 18, 5500000)
emp.display()
```

```
Name: Shruti
Age: 18
Salary: 5500000
```

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

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

0.1.18 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.

[31]:
```python
from abc import ABC, abstractmethod #Abstract Base Class(Module)

class Shape(ABC):
    @abstractmethod
    def draw(self):
        pass

class Rectangle(Shape):
    def draw(self):
        print("Drawing a Rectangle")

class Circle(Shape):
    def draw(self):
        print("Drawing a Circle")

class Triangle(Shape):
    def draw(self):
        print("Drawing a Triangle")

shapes = [Rectangle(), Circle(), Triangle()]

for shape in shapes:
    shape.draw()
```

```
Drawing a Rectangle
Drawing a Circle
Drawing a Triangle
```

[ ]: