Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 1

Keval Dhandhukiya

23010101064

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 [3]:
```
a=10
b=20
print("addition is :",a+b)
```

```
addition is : 30
```

03) WAP to check the type of the variable.

In [5]:
```
print(type(a))
```

```
<class 'int'>
```

04) WAP to calculate simple interest.

In [7]:
```
p = float(input("ENTER NUMBER"))
r = float(input("ENTER NUMBER"))
n = float(input("ENTER NUMBER"))
print((p*r*n)/100)
```

```
0.24
```

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

In [9]:
```
r = float(input("ENTER NUMBER"))
print(3.14*r*r)
```

```
50.24
```

06) WAP to calculate area of a triangle.

In [11]:
```
h = float(input("ENTER NUMBER"))
b = float(input("ENTER NUMBER"))
print(0.5*h*b)
```

```
6.0
```

07) WAP to compute quotient and remainder.

In [13]:
```
a = int(input("ENTER NUMBER"))
b = int(input("ENTER NUMBER"))
c=a/b
x=int(c)
```

```
r = a%b
print(x)
print(r)
```

```
0
6
```

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

In [15]:
```
a = int(input("ENTER NUMBER"))
b = int(input("ENTER NUMBER"))
c = (a*(9/5)+32)
d = ((b-32)*(5/9))
print(c)
print(d)
```

```
33.8
-16.11111111111111
```

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

In [17]:
```
a = int(input("ENTER NUMBER"))
b = int(input("ENTER NUMBER"))
c = int(input("ENTER NUMBER"))
d = int(input("ENTER NUMBER"))
x = ((((b-a)*2)+((d-c)*2))**0.5)
print(x)
```

```
2.449489742783178
```

10) WAP to print sum of n natural numbers.

In [19]:
```
n = int(input("ENTER NUMBER"))
x = (n*(n+1)/2)
print(x)
```

```
3.0
```

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

In [21]:
```
n = int(input("ENTER NUMBER"))
x = ((n*(n+1)*(2*n+1))/6)
print(x)
```

```
30.0
```

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

In [31]:
```
a="eladldkjlk"
b="nsncmxn"
print(a+b)
```

```
eladldkjlknsncmxn
```

13) WAP to swap two numbers.

In [25]:
```
n = int(input("ENTER NUMBER"))
m = int(input("ENTER NUMBER"))
t=n
n=m
m=t
print("n",n)
print("m",m)
```

```
n 3
m 2
```

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

In [27]:
```
km = int(input("ENTER NUMBER"))
m = (km*1000)
feet = (km*3280.84)
inches = (km*39370.1)
cm = (km*100000)
print(km)
print(m)
print(feet)
print(inches)
```

```
print(cm)
```

```
5
5000
16404.2
196850.5
500000
```

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

In [29]:
```
day = int(input("ENTER NUMBER"))
month = int(input("ENTER NUMBER"))
year = int(input("ENTER NUMBER"))
print(day,month,year,sep="-")
```

```
10-2-4
```

In [ ]:
```

```

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 2

Keval Dhandhukiya

23010101064

if..else..

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

In [1]:
```
n = int(input("Enter Number:"))
if(n>0):
     print("Possitive")
else:
     print("Negative")
```

```
Negative
```

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

In [3]:
```
n = int(input("Enter Number:"))
if(n%2==0):
     print("Even")
else:
     print("Odd")
```

```
Even
```

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

In [5]:
```
num1 = int(input("Enter Number:"))
num2 = int(input("Enter Number:"))
if(num1>num2):
    print(f"Number 1 is Large:{num1}")
else:
    print(f"Number 2 is Large:{num2}")
```

```
Number 2 is Large:3
```

In [9]:
```
num1 = int(input("Enter Number:"))
num2 = int(input("Enter Number:"))
print(f"Number 1 is Large:{num1}") if num1>num2 else print(f"Number 2 is Large:{num1}")
```

```
Number 2 is Large:4
```

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

In [11]:
```
num1 = int(input("Enter 1st Number:"))
num2 = int(input("Enter 2nd Number:"))
num3 = int(input("Enter 3rd Number:"))
if(num1>=num2):
    if(num1>=num3):
        print(f"Number 1 is Large:{num1}")
    else:
        print(f"Number 3 is Large:{num3}")
```

```
else:
    if(num2>=num3):
        print(f"Number 2 is Large:{num2}")
    else:
         print(f"Number 3 is Large:{num3}")
```

```
Number 3 is Large:4
```

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 [15]:
```
year = int(input("Enter Year:"))
if(year%4==0):
    if(year%100==0):
        if(year%400==0):
            print(f"Year is Leap Year:{year}")
        else:
            print(f"Year is Not Leap Year:{year}")
    else:
        print(f"Year is Leap Year:{year}")
else:
    print(f"Year is not Leap Year:{year}")
```

```
Year is not Leap Year:2005
```

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

In [1]:
```
day = int(input("Enter 1-7 Number:"))
if(day==1):
    print(f"{day}:Monday")
elif(day==2):
    print(f"{day}:Tuesday")
elif(day==3):
    print(f"{day}:Wednesday")
elif(day==4):
    print(f"{day}:Thursday")
elif(day==5):
    print(f"{day}:Friday")
elif(day==6):
    print(f"{day}:saturday")
elif(day==7):
    print(f"{day}:Sunday")
else:
    print("Enter Proper Day")
```

```
4:Thursday
```

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

In [3]:
```
num1 = int(input("Enter 1st Number:"))
num2 = int(input("Enter 2nd Number:"))
choice = int(input("Enter Choice 1.Add 2.sub 3.Mul 4.Div"))
if(choice==1):
    print(f"Add:{num1+num2}")
elif(choice==2):
    print(f"Sub:{num1-num2}")
elif(choice==3):
    print(f"Mul:{num1*num2}")
elif(choice==4):
    print(f"Div:{num1/num2}")
else:
    print("Enter Proper Number")
```

```
Mul:10
```

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 [5]:
```
pysics = int(input("Enter Pysics Mark:"))
math = int(input("Enter Math Mark:"))
ds = int(input("Enter Ds Mark:"))
python = int(input("Enter python Mark:"))
java = int(input("Enter java Mark:"))
pre = ((pysics + math + ds + python + java) / 500)*100
if(pre>70):
    print(f"Distinction:{pre}")
elif(pre>60 and pre<70):
    print(f"First Calss:{pre}")
```

```
elif(pre>45 and pre<60):
    print(f"Second Calss:{pre}")
elif(pre>35 and pre<45):
    print(f"PASS Calss:{pre}")
else:
    print(f"Fail:{pre}")
```

```
Distinction:74.6
```

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 [7]:
```
a = float(input("Enter the first side of the triangle: "))
b = float(input("Enter the second side of the triangle: "))
c = float(input("Enter the third side of the triangle: "))

if a + b > c and a + c > b and b + c > a:
    if a == b == c:
        print("Equilateral triangle")
    elif a == b or b == c or a == c:
        print("Isosceles triangle")
    elif (a**2 + b**2 == c**2) or (b**2 + c**2 == a**2) or (a**2 + c**2 == b**2):
        print("Right-angled triangle")
    else:
        print("Scalene triangle")
else:
    print("The given sides do not form a valid triangle")
```

```
Scalene triangle
```

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

In [9]:
```
num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))
num3 = int(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)
```

```
The second largest number is: 3
```

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 [15]:
```
unit = int(input("Enter Electric Unit:"))
if(unit>1 and unit<50):
    ans = unit * 2.60
elif(unit<100):
    ans = 130 + ((unit - 50) * 3.25)
elif(unit<=200):
    ans = 292.5 + ((unit - 100) * 5.26)
else:
    ans = 818.5 + ((unit - 200) * 8.45)
    print(f"Ans:{ans}")
```

```
Ans:3353.5
```

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 3

Keval Dhandhukiya

23010101064

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 [1]:
```
n = int(input())
for i in range(1,n+1):
    print(i)
```

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

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

In [27]:
```
num = int(input())
for i in range(1,n+1):
    if(i%2==1):
        print(i)
```

```
 6
1
3
5
7
9
```

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

In [11]:
```
num1 = int(input())
num2 = int(input())
for i in range (num1,num2):
    if(i%2==0 and i%3!=0):
        print(i)
```

```
 4
 6
4
```

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

In [21]:
```
num3 = int(input())
sum = 0
for i in range(1,num3+1):
    sum = sum+i
    print(sum)
```

```
 5
1
3
6
10
15
```

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

In [29]:
```
num4 = int(input())
sum = 0
for i in range(1,num4+1):
    multi = i*i
    sum = sum+multi
print(sum)
```

```
 5
55
```

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

In [33]:
```
num5 = int(input())
sum = 0
for i in range(1,num4+1):
    if(i%2==1):
        sum = sum+i
    elif(i%2==0):
        sum = sum-i
print(sum)
```

```
 4
3
```

08) WAP to print multiplication table of given number.

In [35]:
```
num6=int(input())
for i in range (1,11):
    print(num6,end="*")
    print(i,end="=")
    print(i*num6)
```

```
 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.

In [43]:
```
num7=int(input())
factorial=1
for i in range (1,num7+1):
    factorial=factorial*i
print(factorial)
```

```
 6
720
```

10) WAP to find factors of the given number.

In [41]:
```
num8=int(input())
for i in range(1,num8+1):
    if(num8%i==0):
        print(i,end=",")
```

```
 5
1,5,
```

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

In [55]:
```
num9=int(input())
count = 0
for i in range(1,num9+1):
    if(num9%i==0):
        count=count+1
if(count==2):
    print(num9,"no. is prime")
else:
    print(num9,"no. is mot prime")
```

```
 9
9 no. is mot prime
```

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

In [59]:
```
num10=int(input())
sum=0
while(num10>0):
    remainder=num10%10
    sum=sum+remainder
    num10//=10
print(sum)
```

```
 456
15
```

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

In [5]:
```
palindrome=int(input())
ans=0
num11=palindrome
while(num11!=0):
    remainder=num11%10
    num11=int(num11/10)
    ans = ans*10 + remainder
if(palindrome==ans):
    print(ans,"no. is palindrome")
else:
    print(ans,"no. is not palindrome")
print(ans)
```

```
 121
121 no. is palindrome
121
```

14) WAP to print GCD of given two numbers.

In [1]:
```
x =int(input("enter x:"))
y =int(input("enter y:"))
n = min(x,y)

gcd = 0
for i in range(1,n+1):
    if (x%i == 0 and y%i == 0):
        gcd = i

print(f"GCD of {x} and {y} is {gcd}")
```

```
enter x: 4
enter y: 6
GCD of 4 and 6 is 2.
```

In [ ]:

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 4

Keval Dhandhukiya

23010101064

String

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

In [11]:
```
print("enter the string")
insert = input()
in1 = insert[::-1]
if(in1==insert):
    print("string is palindrome")
else:
    print("string is not palindrome")
```

```
enter the string
 nayan
string is palindrome
```

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

In [15]:
```
print("enter the string")
insert = input()
in2 = insert[::-1]

print("reverse string is:",in2)
```

```
enter thr string
 java
rverse string is: avaj
```

03) WAP to remove ith character from given string.

In [3]:
```
print("enter thr string")
in3 = input()

l = len(in3)
print("enter the ith index no.")
ith = int(input())
in4 =""
for i in range(0,l):

    if(i!=ith):
        in4=in4+in3[i]

print(in4)
```

```
enter thr string
 vadodara
enter the ith index no.
 3
vaddara
```

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

In [26]:
```
print("enter thr string")
insert = input()

count=0
for i in insert:
    count=count+1
print(count)
```

```
enter thr string
 india
5
```

05) WAP to print even length word in string.

In [34]:
```
print("enter thr string")
insert2 = input()

l = len(insert2)
str2 = ""

for i in range(0,l):

    if(i%2==0):
        str2=str2+insert2[i]

print(str2)
```

```
enter thr string
 rajkot
rjo
```

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

In [13]:
```
print("enter thr string")
insert3 = input()
count=0
l = len(insert3)

for i in range(0,l):

    if(insert3[i]=='a' or insert3[i]=='e' or insert3[i]=='i' or insert3[i]=='o' or insert3[i]=='u'):
        count = count+1
print(count)
```

```
enter thr string
 surat
2
```

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

In [1]:
```
s = input("Enter a sentence : ")
words = s.split()
word = []
result = ""
for i in words:
    if len(i)>1:
        i = i[0].upper()+i[1:-1]+i[-1].upper()
    else:
        i=i.upper()
    word.append(i)
result = " ".join(word)
print(result)
```

```
Enter a sentence :  python is very easy
PythoN IS VerY EasY
```

08) WAP to convert given array to string.

In [7]:
```
arr = ["Welcome to python"]
s = ''.join(arr)
print(s)
```

```
Welcome to python
```

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

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

In [9]:
```
password = input("Enter password : ")
confirm = input("Re-enter your password : ")
if password==confirm:
    print("Password Match.")
elif password.lower()==confirm.lower():
    print("Password do not match due to case difference.")
else:
    print("Password do not match.")
```

```
Enter password :  45678
Re-enter your password :  45678
Password Match.
```

10) : Display credit card number.

card no. : 1234 5678 9012 3456

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

In [13]:
```
card_number = input("Enter credit card number : ")
parts = card_number.split()
for i in range(0,len(parts)-1):
    parts[i] = '****'
card_number = " ".join(parts)
print(card_number)
```

```
Enter credit card number :  2314 2548 2546 2548
**** **** **** 2548
```

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

s1 = decimal and s2 = medical are Anagram

In [15]:
```
s1 = input("Enter first string : ")
s2 = input("Enter second string : ")
if sorted(s1)==sorted(s2):
    print("Is Anagram")
else:
    print("Not Anagram")
```

```
Enter first string :  python
Enter second string :  java
Not Anagram
```

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

input : EHlsarwiwhtwMV

output : lsarwiwhtwEHMV

In [21]:
```
s = input("Enter a string : ")
lower = ""
upper = ""
for i in s:
    if i.islower():
        lower += i
    elif i.isupper():
        upper += i
result = lower+upper
print(result)
```

```
Enter a string :   EHlsarwiwhtwMV
lsarwiwhtwEHMV
```

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 5

Keval Dhandhukiya

23010101064

List

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

In [11]:
```
l1 = [1,4,6,8]
sum = 0
for i in range(0,4):
    sum = sum + l1[i]
    print(sum)
```

```
1
5
11
19
```

02) WAP to find largest element in a List.

In [21]:
```
l2 = [2,5,7,8]
max = l2[0]
for i in range(0,4):

    if(max<l2[i]):
        max = l2[i]
print(max)
```

```
8
```

03) WAP to find the length of a List.

In [7]:
```
length = 0
for i in l2:
    length = length + 1
print("length of list :",length)
```

```
length of list : 4
```

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

In [13]:
```
l1[0], l1[length - 1] = l1[length-1], l1[0]
print("1st index",l1[0],"last index",l1[length - 1])
```

```
1st index 8 last index 1
```

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

In [61]:
```
l3 = [2,4,6,7]
l4 = []
l5 = []
```

```
for i in range(0,4):
    if(i > (4/2)):
        l4.append(l3[i])
    if(i < (4/2)):
        l5.append(l3[i])
print(l3)

print("alter split 1st list is l4 :",l4,"2nd list is :",l5)
l6 = [l4,l5]
print(l6)
```

```
[2, 4, 6, 7]
alter split 1st list is l4 : [7] 2nd list is : [2, 4]
[[7], [2, 4]]
```

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

In [27]:
```
print("Enter an index :")
i1 =int(input())
i2 =int(input())

l2[i1],l2[i2] = l2[i2],l2[i1]
print("At",i1,"th index :",l2[i1],"and at",i2,"th index :",l2[i2])
```

```
Enter an index :
At 1 th index 8 and at 3 th index 5
```

07) WAP to reverse the list entered by user.

In [57]:
```
user = input("enter the list :")
l7 = [int(x) for x in user.split()]

l7.reverse()
print("reverse list :",l7)
```

```
reverse list : [4, 3, 2, 1]
```

08) WAP to print even numbers in a list.

In [63]:
```
l8 = [l2[i] for i in range(0,4) if(l2[i]%2==0)]
print(l8)
```

```
[2, 8]
```

09) WAP to count unique items in a list.

In [13]:
```
input_list = [1, 2, 2, 5, 8, 4, 4, 8]

l1 = []

count = 0

for item in input_list:
    if item not in l1:
        count += 1
        l1.append(item)

print("No of unique items are:", count)
```

```
No of unique items are: 5
```

10) WAP to copy a list.

In [67]:
```
l2.copy()
print(l2)
```

```
[2, 8, 7, 5]
```

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

In [2]:
```
lower = int(input("Enter lower range : "))
upper = int(input("Enter upper range : "))
li = [i for i in range(lower,upper+1) if i%2==1]
print(li)
```

```
[5]
```

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

In [83]:
```
l10 = [1,2,4,6,7,8,9]
```

```
print(l10.count(2))
print(l10.count(3))
```

```
1
0
```

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

In [11]:
```
l11 = [2,4,3,5]
max = l11[0]
secondmax = l11[0]
for i in range(0,4):
    if(max<l11[i]):
        max = l11[i]
for i in range(0,4):
    if(secondmax<l11[i] and l11[i]!=max):
        secondmax = l11[i]
print("1st max no. is :",max,"and 2nd max no. is :",secondmax)
```

```
1st max no. is : 5 and 2nd max no. is : 4
```

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

In [6]:
```
n = input("Enter space separated values :")
li = [i for i in n.split()]
k = int(input("Enter value of k : "))
ans = set([i for i in li if li.count(i)>k])
print(list(ans))
```

```
[]
```

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

In [12]:
```
l12=[]
for i in range(0,10):
    l12.append(i**2)
print("Without List Comprehension",li)

l13 = [i**2 for i in range(0,10)]
print("With list comprehension : ",l13)
```

```
Without List Comprehension ['4', '5', '8', '9']
With list 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 [14]:
```
n = input("Enter space separated fruits : ")
li = [i for i in n.split()]
fruits = [i for i in li if i.startswith("b")]
print(fruits)
```

```
['banana', 'barberry', 'bataun']
```

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

In [18]:
```
n1 = input("Enter space separated values : ")
n2 = input("Enter space separated values : ")
l14 = [i for i in n1.split()]
l15 = [i for i in n2.split()]
common = [x for x in l14 if x in l15]
print(common)
```

```
['4', '5']
```

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 6

Keval Dhandhukiya

23010101064

Tuple

01) WAP to find sum of tuple elements.

In [1]:
```
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 [3]:
```
t = (1,2,3,4,9,6)
t = sorted(t)
k = int(input("enter a number"))

for i in range(0,k+1):
    print(f"min {i} is {t[i]}  max {i} is {t[-i-1]}")
```

```
min 0 is 1  max 0 is 9
min 1 is 2  max 1 is 6
min 2 is 3  max 2 is 4
min 3 is 4  max 3 is 3
min 4 is 6  max 4 is 2
min 5 is 9  max 5 is 1
```

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

In [5]:
```
list = [(4,8,12),(4,8,45),(4,5,7)]
k = int(input("enter number to divide tuple"))
for i in list:
    for j in range(0,len(i)):
        if(i[j]%k!=0):
            break
    else:
        print(i,"is divisible by ",k)
```

```
(4, 8, 12) is divisible by  4
```

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

In [7]:
```
l = [1,2,3,4,5]

l2 = []

for i in l:
    t = (i,i**3)
    l2.append(t)
```

```
print(l2)
```

```
[(1, 1), (2, 8), (3, 27), (4, 64), (5, 125)]
```

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

In [9]:
```
list = [(4,-8,12),(4,8,-45),(4,5,7)]
for i in list:
    for j in range(0,len(i)):
        if(i[j]<0):
            break
    else:
        print(i,"is positive tuple")
```

```
(4, 5, 7) is positive tuple
```

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

In [11]:
```
l = [(1,2,3)]
l.append((4,5,6))
print(l)

t = tuple([1,2,3])
t = t + tuple([4,5,6])
print(t)
```

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

07) WAP to remove tuples of length K.

In [13]:
```
l = [(1,2),(1,2,3),(4,5,6,7),(1,2,3),(7,8)]
k = int(input("enter a number"))
for i in l:
    if(len(i)==k):
        l.remove(i)
print(l)
```

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

08) WAP to remove duplicates from tuple.

In [15]:
```
t  = (1,2,3,1,4,5,6,2,1)
print(t)
s = set(t)
t = tuple(s)
print(t)
```

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

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

In [17]:
```
t = (1,2,3,4,5,6)
l = []
for i in range(0,len(t)-1):
    l.append(t[i]*t[i+1])

tres = tuple(l)
print(tres)
```

```
(2, 6, 12, 20, 30)
```

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

In [19]:
```
t = (1,2,3,4,5,6,7,8,1)
for i in t:
    if(t.count(i)!=1):
        print("given tuple is not distinct")
        break
else:
    print("given tuppel is distinct")
```

```
given tuple is not distinct
```

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 7

Keval Dhandhukiya

23010101064

Set & Dictionary

01) WAP to iterate over a set.

In [15]:
```
s1 = set("Keval")

for val in s1:
    print(val)
```

```
K
e
l
a
v
```

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

In [9]:
```
s2 = {2,4,5,6,7}
new1 = list(s2)
new2 = tuple(s2)
new3 = str(s2)
print(new1,new2,new3)
```

```
[2, 4, 5, 6, 7] (2, 4, 5, 6, 7) {2, 4, 5, 6, 7}
```

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

In [5]:
```
s3 = {5, 2, 8, 1, 9, 2, 18}

s4 = list(s3)

maximum = max(s4)
minimum = min(s4)

print("Maximum:", maximum)
print("Minimum:", minimum)
```

```
Maximum: 18
Minimum: 1
```

04) WAP to perform union of two sets.

In [7]:
```
s5 = {1,4,5,7,8}
s6 = {1,3,5,9,6}
new_set = s5.union(s6)
print(new_set)
```

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

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

In [17]:
```
a = [1, 2, 3, 4]
b = [3, 5, 6]

common = set(a) & set(b)

print(bool(common))
```

```
True
```

06) WAP to remove duplicates from list.

In [19]:
```
c = [1, 2, 2, 3, 4, 4, 5]

unique_list = list(set(c))
print(unique_list)
```

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

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

In [33]:
```
str1 = input("Enter a sentence: ")

print("Total words: ", str1.split())

unique = set(str1)
print("Unique words: ",unique)
```

```
Total words:  ['python', 'is', 'very', 'easy']
Unique words:  {'e', 'r', 'o', 't', 'y', 'p', 'h', 's', 'v', 'a', 'i', 'n', ' '}
```

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

In [25]:
```
d = [1, 2, 3, 4, 5]
e = [4, 5, 6, 7, 8]

result1 = list(set(d) - set(e))
result2 = list(set(e) - set(d))

f = result1 + result2
print(f)
```

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

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

In [37]:
```
def isAnagram(str1, str2):

    if len(str1) != len(str2):
        return False

    if sorted(str1) == sorted(str2):
        return True
    else:
        return False

str1 = "pythontpoint"
str2 = "tnioptnohtyp"

if isAnagram(str1, str2):
    print("Anagrams")
else:
    print("Not Anagrams")
```

```
Anagrams
```

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

In [39]:
```
myArr1 = [5, 10, 15, 20, 25]
myArr2 = [2, 5, 6, 7, 10, 15, 18, 20]
myArr3 = [10, 20, 30, 40, 50, 60]

print("First = ",myArr1)
print("Second = ",myArr2)
print("Third = ",myArr3)

output_set = set(myArr1) & set(myArr2) & set(myArr3);

final_list = list(output_set)
```

```
print("Common elements = ",final_list)
```

```
First =  [5, 10, 15, 20, 25]
Second =  [2, 5, 6, 7, 10, 15, 18, 20]
Third =  [10, 20, 30, 40, 50, 60]
Common elements =  [10, 20]
```

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

In [43]:
```
string = "GeekforGeeks!"
vowel = "aeiouAEIOU"

count = sum(string.count(vowel) for vowel in vowel)
print(count)
```

```
5
```

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

In [49]:
```
string1 = "01010101010"
if(string1.count('0')+string1.count('1')==len(string1)):
        print("Yes")
else:
        print("No")
```

```
Yes
```

13) WAP to sort dictionary by key or value.

In [51]:
```
d = {'ravi': 10, 'rajnish': 9, 'sanjeev': 15}

myKeys = list(d.keys())
myKeys.sort()

sd = {i: d[i] for i in myKeys}
print(sd)
```

```
{'rajnish': 9, 'ravi': 10, 'sanjeev': 15}
```

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

In [53]:
```
def returnSum(myDict):

        list = []
        for i in myDict:
                list.append(myDict[i])
        final = sum(list)

        return final

dict = {'a': 100, 'b': 200, 'c': 300}
print("Sum :", returnSum(dict))
```

```
Sum : 600
```

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 [57]:
```
dict1 = {'a': 5, 'c': 8, 'e': 2}
if "q" in dict1:
        print(dict1["d"])
else:
        print("Key not found")
```

```
Key not found
```

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 8

Keval Dhandhukiya

23010101064

User Defined Function

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

In [1]:
```
print("enter the mass:")
mass = int(input())

print("enter the height:")
height = int(input())

def calculate_BMI(mass,h):
    print("BMI = ",(mass/(h**2)))

calculate_BMI(mass,height)
```

```
enter the mass:
enter the height:
BMI =  0.1388888888888889
```

02) Write a function that add first n numbers.

In [7]:
```
print("enter the number:")
n = int(input())

def sum1(number):
    sum = 0

    for i in range(0,n+1):
        sum += i

    print(sum)
sum1(number)
```

```
enter the number:
10
```

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

In [9]:
```
def is_prime(number):
    if number <= 1:
        return 0
    for i in range(2, int(number ** 0.5) + 1):
        if number % i == 0:
            return 0
    return 1
num = 7
result = is_prime(num)
print(f"Is {num} a prime number? {'Yes' if result == 1 else 'No'}")
```

```
Is 7 a prime number? Yes
```

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

In [11]:
```
print("enter the starting range:")
number2 = int(input())

print("enter the ending range:")
number3 = int(input())

def prime(x, y):
    prime_list = []
    for i in range(x, y):
        if i == 0 or i == 1:
            continue
        else:
            for j in range(2, int(i/2)+1):
                if i % j == 0:
                    break
            else:
                prime_list.append(i)
    return prime_list

prime(number2,number3)
```

```
enter the starting range:
enter the ending range:
```

Out[11]:
```
[2, 3, 5, 7]
```

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

In [13]:
```
def isPalindrome(s):
    return s == s[::-1]

s = "nayan"
ans = isPalindrome(s)

if ans:
    print("True")
else:
    print("False")
```

```
True
```

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

In [15]:
```
a = [10, 20, 30, 40]
res = 0

for val in a:

    res += val

print(res)
```

```
100
```

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

In [17]:
```
t1 = (2,4,6)
t2 = (3,5,7)

l1 = (t1,t2)

def summation(l1):

        sum = 0

        for i,j,k in l1:
                sum += i
        return sum

summation(l1)
```

Out[17]:
```
5
```

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

In [19]:
```
print("enter an value:")
n1 = int(input())
```

```
def nth_fibonacci(n):

    if n <= 1:
        return n

    return nth_fibonacci(n - 1) + nth_fibonacci(n - 2)

result = nth_fibonacci(n1)
print(result)
```

```
enter an value:
8
```

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 [21]:
```
def get_student_name(rollno, student_dict):
    return student_dict.get(rollno, "Student not found")

dict1 = {101: 'Ajay', 102: 'Rahul', 103: 'Jay', 104: 'Pooja'}
rollno = 103

student_name = get_student_name(rollno, dict1)
print(f"The name of the student with roll number {rollno} is {student_name}.")
```

```
The name of the student with roll number 103 is Jay.
```

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 [23]:
```
scores = [200, 456, 300, 100, 234, 678]

def fn10(scores):
    sum1 = 0
    for i in scores:
        if(i%10 == 0):
            sum1 += i
    return sum1

print(fn10(scores))
```

```
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 [25]:
```
def invert_dictionary(original_dict):
    return {value: key for key, value in original_dict.items()}

original_dict = {'a': 10, 'b': 20, 'c': 30, 'd': 40}
inverted_dict = invert_dictionary(original_dict)
print("Original Dictionary:", original_dict)
print("Inverted Dictionary:", inverted_dict)
```

```
Original Dictionary: {'a': 10, 'b': 20, 'c': 30, 'd': 40}
Inverted Dictionary: {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 [27]:
```
def is_pangram(input_string):
    alphabet = set('abcdefghijklmnopqrstuvwxyz')
    input_characters = set(input_string.lower())
    return alphabet.issubset(input_characters)

test_string = "The quick brown fox jumps over the lazy dog"
if is_pangram(test_string):
    print(f'"{test_string}" is a pangram.')
```

```
else:
    print(f'"{test_string}" is NOT a pangram.')
```

```
"The quick brown fox jumps over the lazy dog" is a pangram.
```

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 [29]:
```
def fn13(s):
    dict1 = {"upper":0,"lower":0}
    for i in s:
        if(i.islower()):
            dict1["lower"]+=1
        else:
            dict1["upper"]+=1
    print(f"no_upper={dict1["upper"]},no_lower={dict1["lower"]}")

s1 = "AbcDEfgh"
fn13(s1)
```

```
no_upper=3,no_lower=5
```

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

In [31]:
```
ans = lambda x,y : x if x>y else y
print(ans(10,20))
```

```
20
```

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

In [33]:
```
l = ["Keval","Karan","Darshan","Ironman","Superman","Homeleander"]

l2 = list(filter(lambda i : len(i)>7,l))

print(l2)
```

```
['Superman', 'Homeleander']
```

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

In [35]:
```
l = ["keval","karan","darshan","ironman","superman"]

l = list(map(lambda i : i.capitalize(),l))

print(l)
```

```
['Keval', 'Karan', 'Darshan', 'Ironman', 'Superman']
```

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 [37]:
```
#position argument
# def add(a,b):
#     return a+b
# print(add(10,20))


#Keyword Argument
# def add3(a,b,c):
#     return a+b+c

# print(add3(b=10,c=20,a=5))
#print(add3(b=10,c=20,10))  it's not working



#default argument
# def add4(a,b,c,d=10):
#     return a+b+c+d
```

```
# print(add4(10,20,30,40))
# print(add4(10,20,30))


# *args
# def add5(*args):
#     sum1 = 0
#     for i in args:
#         sum1 += i
#     return sum1
# print(add5(1,2,3,4,5))


# **kwargs
# def addstu(**kwargs):
#     for i in kwargs.items():
#         print(i)

# addstu(Keval=384,Chirag=373,Harsh=123,Darshan=245)

#only keyword
#def fn2(*,a,b,c):
#    print(a,b,c,sep=" ")
#fn2(b=10,a=2,c=3)

#only positional

def fn(a,b,c,/):
    print(a,b,c,sep=" ")
fn(1,2,3)
```

```
1 2 3
```

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 9

Keval Dhandhukiya

23010101064

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 [5]:
```
fp = open("KD.txt","r")

str1 = fp.read()
print(str1)

line1 = fp.readline()
print(line1)

l1 = []

l1 = fp.readline()
print(l1)
```

```
Keval Dhandhukiya

Darshan University
```

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

In [47]:
```
fp = open("new.txt","w")
fp.close()
```

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

In [31]:
```
fp = open("KD.txt","r")
c = 0
d = fp.readline()
while(c<5):
    print(d, end = "")
    d = fp.readline()
    c+=1
fp.close()
```

```
Keval
Dhandhukiya
Darshan
University
Rajkot
```

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

In [35]:
```
fp = open("KD.txt","r")
wl = fp.read().split()
w_len = list(map(len,wl))
maxi = max(w_len)
ans = [i for i in wl if len(i) == maxi]
print(ans)
fp.close()
```

```
['Dhandhukiya']
```

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

In [37]:
```
fp = open("KD.txt","r")
l = 0
w = 0
c = 0
for i in fp:
    l += 1
    for j in i.split():
        w += 1
        for k in j:
            c += 1
print(f"Lines {l}, Words {w}, characters {c}")
fp.close()
```

```
Lines 7, Words 7, characters 48
```

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

In [45]:
```
fp = open("KD.txt","r")
data = fp.read()
fw = open("Copy1.txt","w")
fw.write(data)
fp.close()
fw.close()
```

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

In [49]:
```
import os
s = os.path.getsize("KD.txt")
print("size of this file : ",s,"B")
```

```
size of this file :  62 B
```

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

In [55]:
```
word = input("Enter words:")
fp = open("KD.txt","r")
def frequency(f,w):
    c = 0
    for i in f:
        for j in i.split():
            if(j == w):
                c += 1
    return c
print(frequency(fp,word))
fp.close()
```

```
Enter words: Keval
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 [59]:
```
d={"A":90,"B":99,"C":100,"D":96,"E":20}
m=0
f=open("Marks.txt","x")
for i in d.keys():
    if(d[i]>m):
        m=d[i]
```

```
    f.write(f"{i} : {d[i]} \n")
f.close()

print(m)
```

```
100
```

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

(Note: each number should be in new line)

In [4]:
```
fp = open("primenumbers.txt", 'w')

primes = []
number = 2

while len(primes) < 100:
    is_prime = True
    for i in range(2, int(number**0.5) + 1):
        if number % i == 0:
            is_prime = False
            break

    if is_prime:
        primes.append(number)
        fp.write(f"{number}\n")

    number += 1

fp.close()
```

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

In [63]:
```
f1=open("KD.txt")
d1=f1.read()
f1.close()

f2=open("Marks.txt")
d2=f2.read()
f2.close()

f=open("KD2.txt","x")
f.write(d1+d2)
f.close()
```

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

In [67]:
```
word1 = "oldword"
word2 = "newword"

fp = open("input.txt", 'r')
data = fp.read()
fp.close()

updated_data = data.replace(word1, word2)

fp = open("output.txt", 'w')
fp.write(updated_data)
fp.close()

print(f"Replaced '{word1}' with '{word2}'")
```

```
Replaced 'oldword' with 'newword'
```

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

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 10

Keval Dhandhukiya

23010101064

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 [1]:
```
try:
    print(10/0)
except ZeroDivisionError as arr:
    print(arr)
```

```
division by zero
```

In [3]:
```
import math
b = -18
try:
    print(math.sqrt(b))
except ValueError as arr:
    print(arr)
```

```
math domain error
```

In [7]:
```
a = "keval"
b = 18
try:
    print(a+b)
except TypeError as arr:
    print(arr)
```

```
can only concatenate str (not "int") to str
```

02) WAP to handle following exceptions:

1. IndexError
2. KeyError

In [11]:
```
lt = [12,23,4,5,76,78]
try:
    print(lt[6])
except IndexError as err:
    print(err)
```

```
list index out of range
```

In [13]:
```
dr = { 1 : "keval", 2 : "Akhil" }
try:
    print(dr[3])
```

```
except KeyError as err:
    print("Key Not Found")
```

```
Key Not Found
```

03) WAP to handle following exceptions:

1. FileNotFoundError
2. ModuleNotFoundError

In [30]:
```
try:
    fp = open("abc.txt","r")
except FileNotFoundError as err:
    print(err)
```

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

In [28]:
```
try:
    import keval
except ModuleNotFoundError as err:
    print(err)
```

```
No module named 'keval'
```

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

In [36]:
```
try:
    #fp = open("abc.txt","r")
    print(10/0)
except Exception as err:
    print(err)
```

```
division by zero
```

05) WAP to demonstrate else and finally block.

In [38]:
```
try:
    #import keval
    print(18)
except ModuleNotFoundError as err:
    print(err)
else:
    print("keval is busy")
```

```
18
keval is busy
```

In [50]:
```
try:
    import keval
except ModuleNotFoundError as err:
    print(err)
else:
    print("not coming")
finally:
    print("keval is busy")
```

```
No module named 'keval'
keval is busy
```

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 [52]:
```
li = list(input("Enter Grades coma saperated : ").split(","))
try:
    li2 = list(int(i for i in li))
except TypeError as err :
    print(err)
else:
    print(li)
```

```
int() argument must be a string, a bytes-like object or a real number, not 'generator'
```

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

```

```

In [54]:
```
def division(a,b):
    try:
        print("division : ",a//b)
    except ZeroDivisionError as err :
        print(err)

#division(10,5)
division(10,0)
```

```
integer division or modulo by zero
```

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 [3]:
```
age = int(input("Enter Age : "))

try:
    if(age<18):
        raise ValueError("Age is not Valid")
    else:
        print("Age is :",age)
except ValueError as err:
    print(err)
```

```
Age is not Valid
```

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 [3]:
```
class InvalidUsernameError(Exception):
    def __init__(self,msg):
        self.msg = msg

name = input("Enter Name : ")
try:
    if(len(name)<5 or len(name)>15 ):
        raise InvalidUsernameError("Username must be between 5 and 15 characters long")
    else:
        print("Name :",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 [7]:
```
class NegativeNumberError(Exception):
    def __init__(self,msg):
        self.msg = msg

num = int(input("Enter number : "))
try:
    if(num<0):
        raise NegativeNumberError("Cannot calculate the square root of a negative number")
    else:
        print("square root is :",math.sqrt(num))
except NegativeNumberError as err:
    print(err)
```

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

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 11

Keval Dhandhukiya

23010101064

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 [3]:
```
import cal as fn
add = fn.add(2,3)
print(add)
```

```
5
```

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

In [5]:
```
import random
str = 'Hello world from darshan university'
ran = random.randint(0,len(str))
print(str[ran])
```

```
a
```

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

In [7]:
```
import random
list = [10,20,30,20,10,20,30,40,50,60,70,30,20,430,233]
ran = random.randint(0,len(list))
print(list[ran])
```

```
30
```

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

In [9]:
```
import random
random.seed(10)
print(int(random.random() * 10))
```

```
5
```

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

In [11]:
```
import random
import math
ran1 = random.randint(100,999)
ran2 = random.randint(100,999)
ran3 = random.randint(100,999)
firstNum = math.ceil(ran1 / 5) * 5
secondNum = math.ceil(ran2 / 5) * 5
thirdNum = math.ceil(ran3 / 5) * 5
print(firstNum, secondNum, thirdNum)
```

```
540 595 695
```

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 [13]:
```
import random
import math
lottery_tickets = []
for i in range(0,100):
    ran = int(random.random() * 10**10)
    lottery_tickets.append(ran)
winner1 = int(random.randint(0,100))
winner2 = int(random.randint(0,100))
print(f"First Winner is {lottery_tickets[winner1]} and Runner up is {lottery_tickets[winner2]}")
```

```
First Winner is 3486810954 and Runner up is 1760894393
```

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

In [15]:
```
import datetime
now = datetime.datetime.now()
print(now)
```

```
2025-02-28 20:21:06.890282
```

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

In [51]:
```
from datetime import datetime, timedelta
now = datetime.now() + timedelta(days=-6)
print(now.date())
```

```
2025-02-22
```

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

In [45]:
```
from datetime import datetime

strdate1 = '23/02/2025'
strdate2 = '21/02/2022'

date1 = datetime.strptime(strdate1, "%d/%m/%Y")
date2 = datetime.strptime(strdate2, "%d/%m/%Y")


print(date1 - date2)
```

```
1098 days, 0:00:00
```

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

In [43]:
```
import datetime

date = datetime.date(2024, 2, 17)

day_name = date.strftime("%A")
sort_day = date.strftime("%a")

print(day_name, sort_day)
```

```
Saturday Sat
```

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

In [19]:
```
from datetime import date

my_date = date(1996, 12, 13)

print("Date is", my_date)

# Uncommenting my_date = date(1996, 12, 39)
# will raise an ValueError as it is
# outside range
```

```
Date is 1996-12-13
```

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

In [25]:
```
import math as m1

print(m1.pi)
print(m1.e)
print (math.cos(0.00))
print(math.sin(0.00))
print(math.atan(0.00))
print(math.tan(1.00))
```

```
3.141592653589793
2.718281828459045
1.0
0.0
0.0
1.5574077246549023
```

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

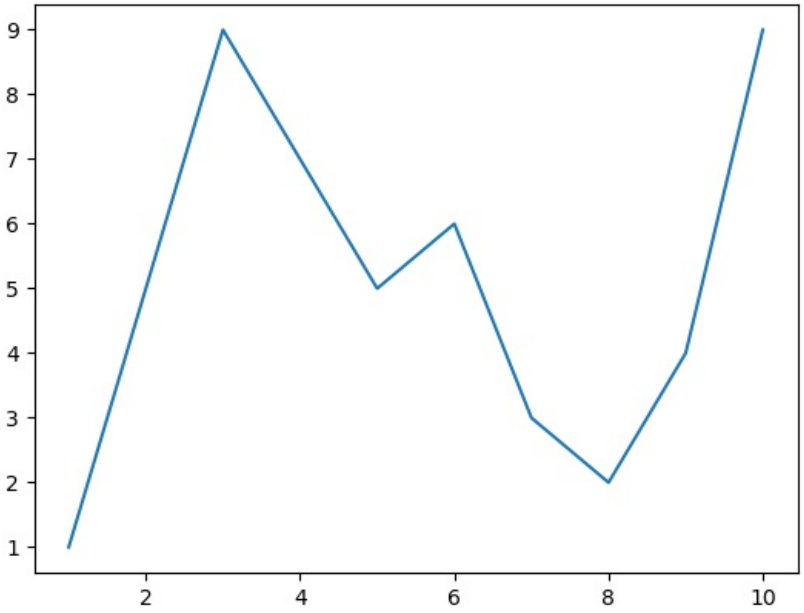
Lab - 12

Keval Dhandhukiya

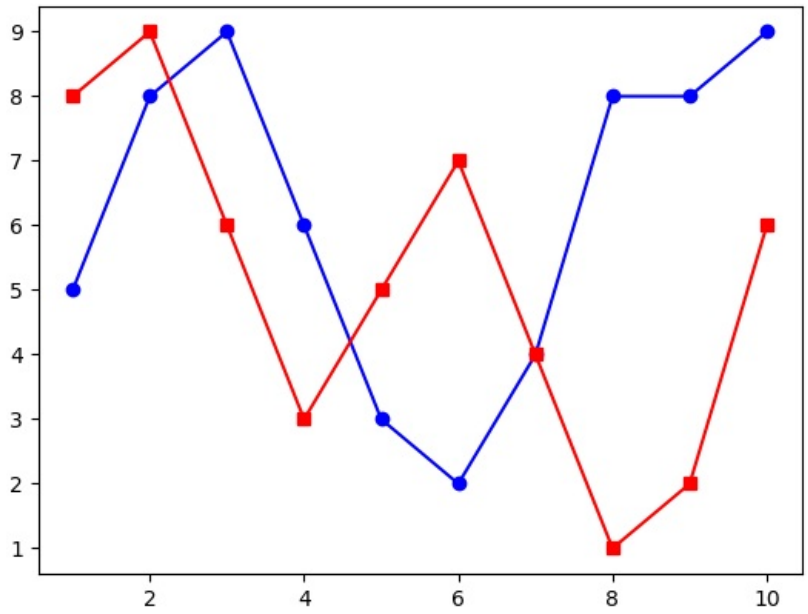
23010101064

In [8]:
```
#import matplotlib below
import matplotlib.pyplot as plt
```

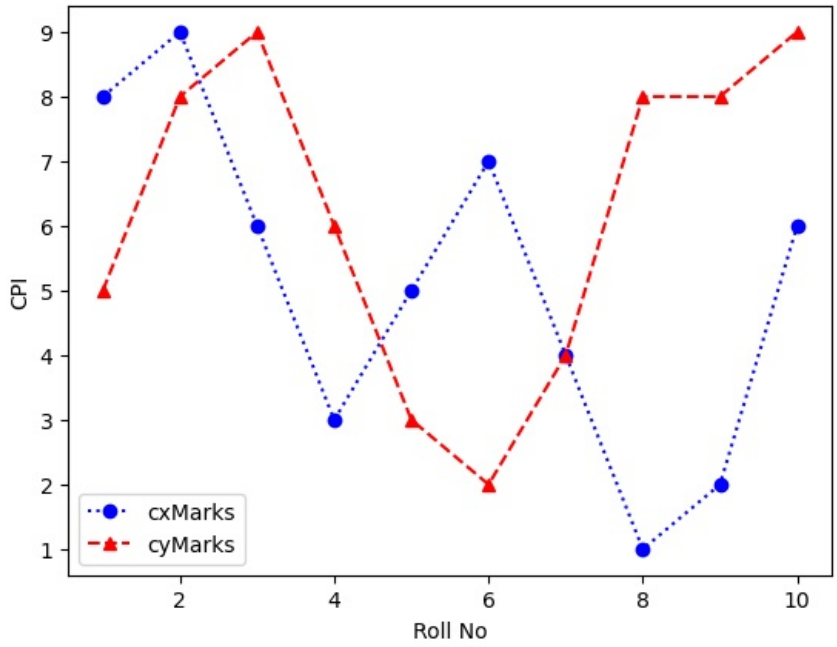
In [10]:
```
x = range(1,11)
y = [1,5,9,7,5,6,3,2,4,9]
plt.plot(x,y)
plt.show()
# write a code to display the line chart of above x & y
```



In [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]
plt.plot(x,cxMarks,label="cxMarks",color="b",marker="o")
plt.plot(x,cyMarks,label="cyMarks",color="r",marker="s")
plt.show()
# write a code to display two lines in a line chart (data given above)
```



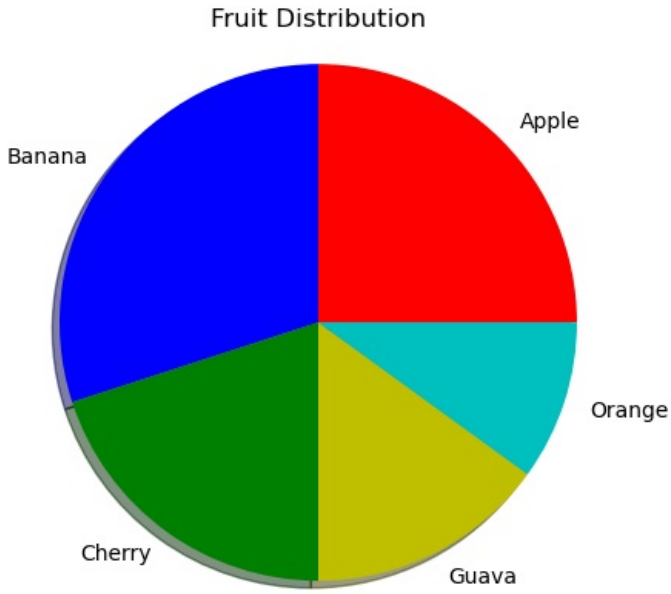
In [14]:
```
x = range(1,11,1)
cxMarks= [8,9,6,3,5,7,4,1,2,6]
cyMarks= [5,8,9,6,3,2,4,8,8,9]
plt.plot(x,cxMarks,label="cxMarks",color="b",marker="o",linestyle="dotted")
plt.plot(x,cyMarks,label="cyMarks",color="r",marker="^",linestyle="dashed")
plt.xlabel("Roll No")
plt.ylabel("CPI")
plt.legend()
plt.show()
# write a code to generate below graph
```



04) WAP to demonstrate the use of Pie chart.

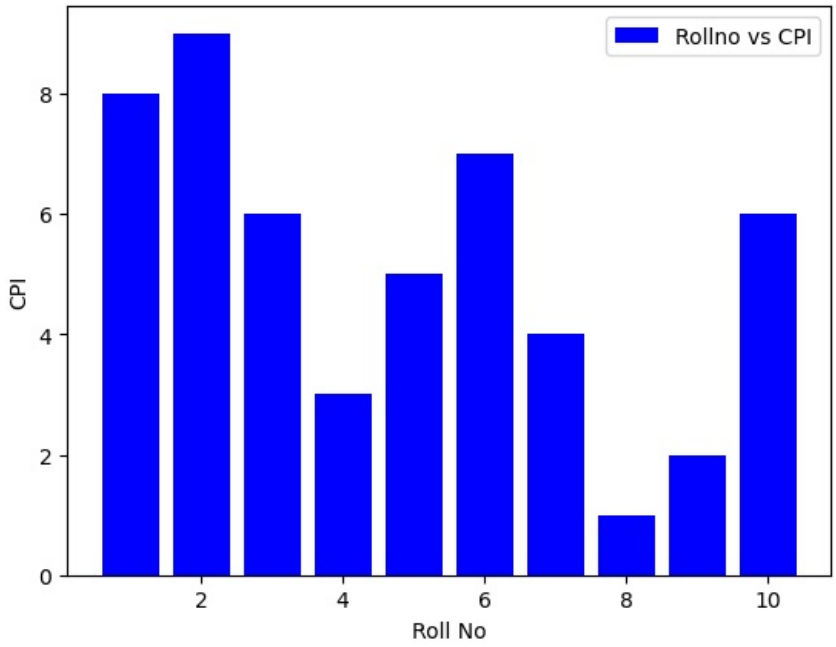
In [44]:
```
labels = ['Apple', 'Banana', 'Cherry', 'Guava', 'Orange']
sizes = [25, 30, 20, 15, 10]
colors = ['r', 'b', 'g', 'y', 'c']

plt.pie(sizes, labels=labels, colors=colors, shadow=True)
plt.axis('equal')
plt.title('Fruit Distribution')
plt.show()
```



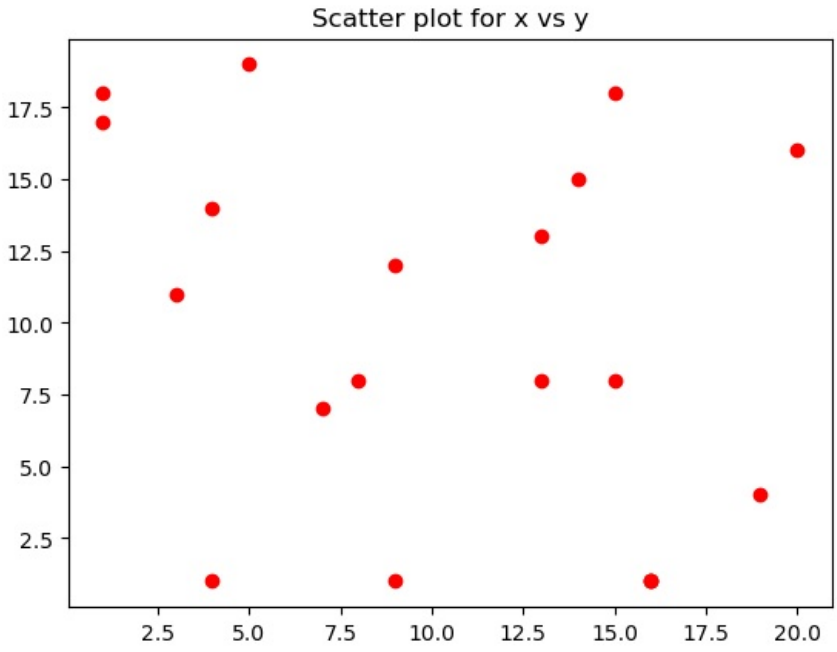
05) WAP to demonstrate the use of Bar chart.

In [18]:
```
x = range(1,11,1)
cxMarks= [8,9,6,3,5,7,4,1,2,6]
plt.bar(x,cxMarks,label="Rollno vs CPI",color="b")
plt.xlabel("Roll No")
plt.ylabel("CPI")
plt.legend()
plt.show()
```



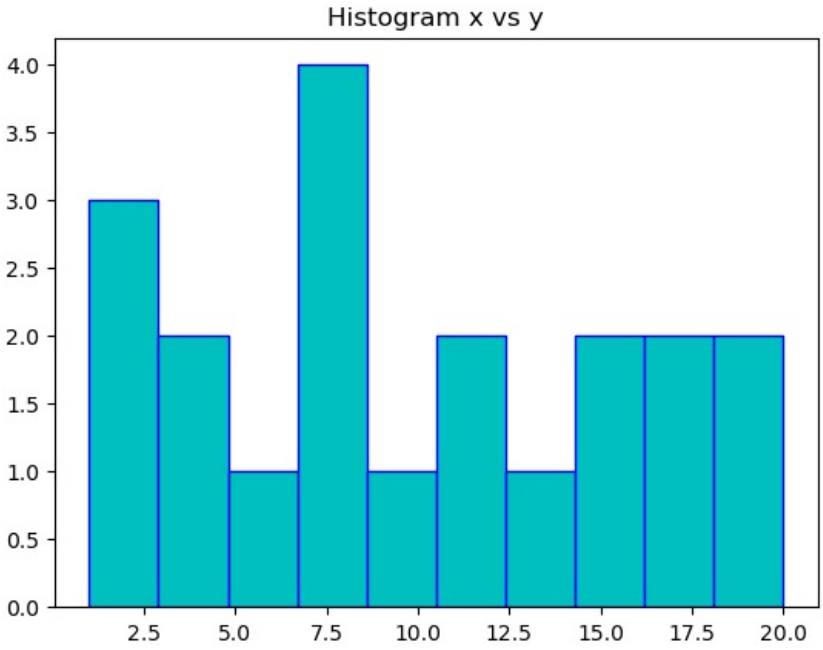
06) WAP to demonstrate the use of Scatter Plot.

In [62]:
```
import random as r
r.seed(1)
x = [r.randint(1,20) for i in range(20)]
y = [r.randint(1,20) for i in range(20)]
plt.scatter(x,y,color="r")
plt.title("Scatter plot for x vs y")
plt.show()
```



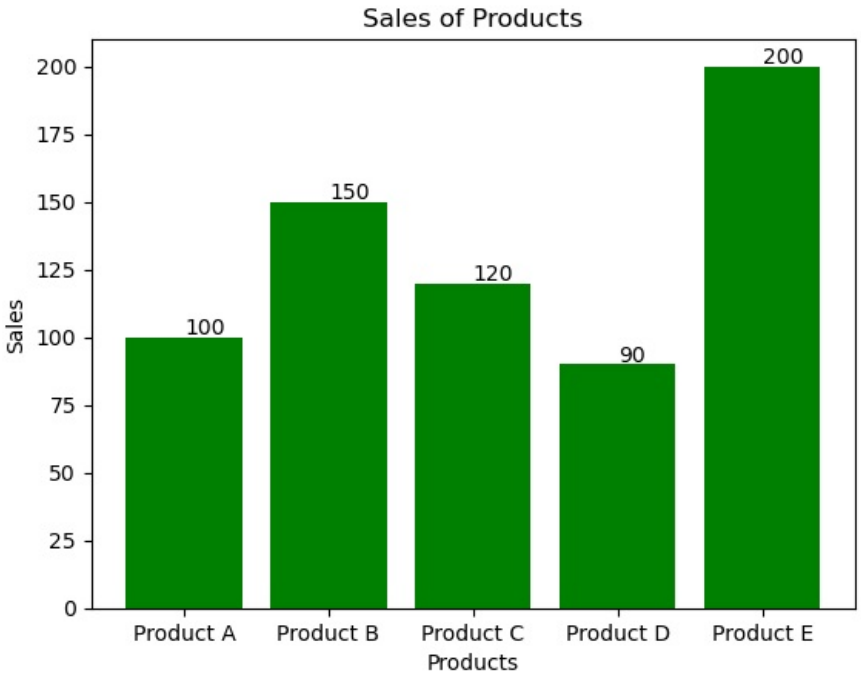
07) WAP to demonstrate the use of Histogram.

In [50]:
```
r.seed(5)
data = [r.randint(1,20) for i in range(20)]
plt.hist(data,edgecolor="b",color="c")
plt.title("Histogram x vs y")
plt.show()
```



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

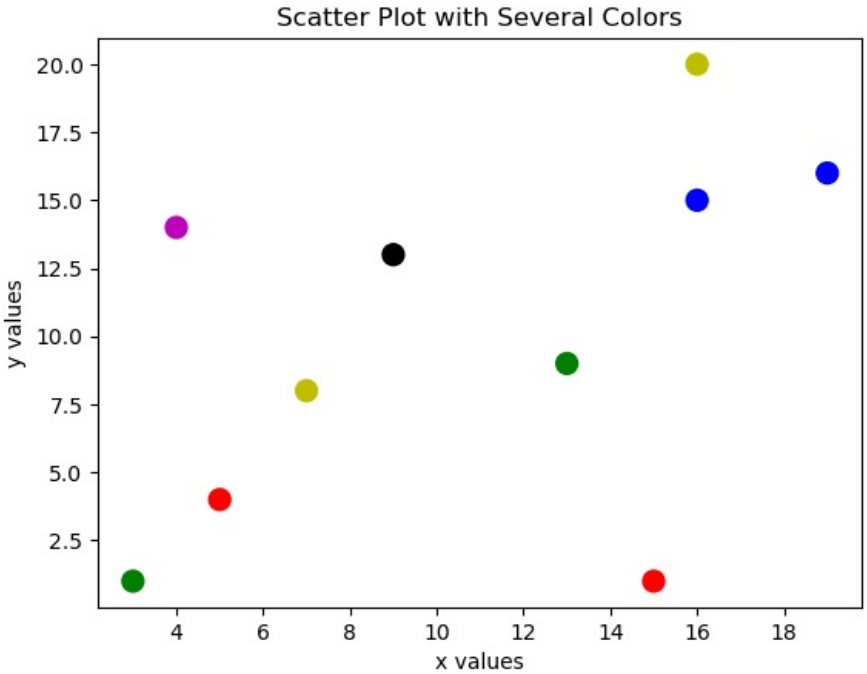
In [56]:
```
products = ['Product A', 'Product B', 'Product C', 'Product D', 'Product E']
sales = [100, 150, 120, 90, 200]
plt.bar(products, sales, color='g')
for i in range(len(products)):
    plt.text(i, sales[i]+1,str(sales[i]))
plt.title('Sales of Products')
plt.xlabel('Products')
plt.ylabel('Sales')
plt.show()
```



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

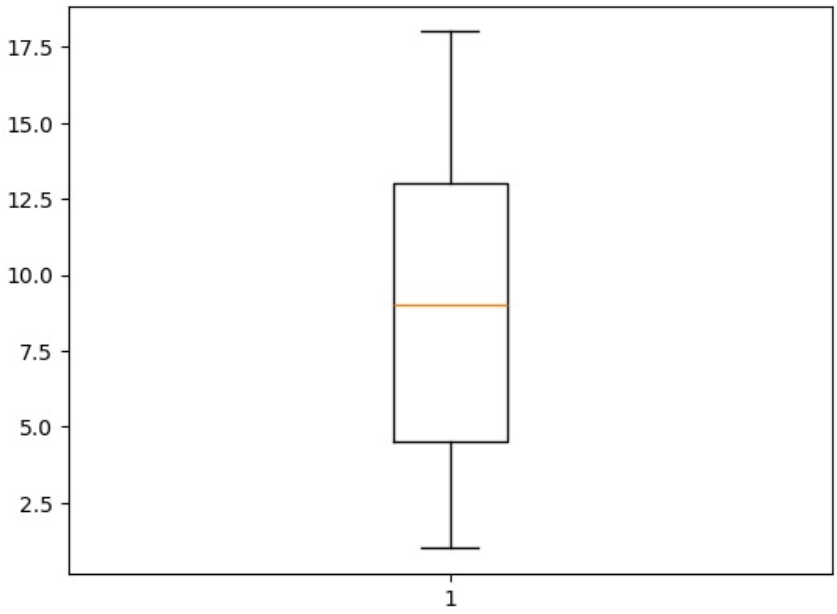
In [58]:
```
r.seed(1)
x = [r.randint(1,20) for i in range(10)]
y = [r.randint(1,20) for i in range(10)]
colors = ['r','b','g','k','m','y','r','b','g','y']

plt.scatter(x, y, color=colors, s=100)
plt.title('Scatter Plot with Several Colors')
plt.xlabel('x values')
plt.ylabel('y values')
plt.show()
```



10) WAP to create a Box Plot.

In [64]:
```
data=[r.randint(1,20) for i in range(10)]
plt.boxplot(data)
plt.show()
```



Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 13

Keval Dhandhukiya

23010101064

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

obj=Student("Keval",19,"A")

print(obj.name)
print(obj.age)
print(obj.grade)
```

```
Keval
19
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 [3]:
```
class Bank_Account:
    def __init__(self):
        pass

    def GetAccountDetails(self):
        self.Account_no=int(input("Enter Account_no:"))
        self.User_Name=input("Enter User_Name:")
        self.Email=input("Enter Email:")
        self.Account_Type=input("Enter Account_Type:")
        self.Account_Balance=int(input("Enter Account_Balance:"))

    def DisplayAccountDetails(self):
        print(self.Account_no)
        print(self.User_Name)
        print(self.Email)
        print(self.Account_Type)
        print(self.Account_Balance)

obj=Bank_Account()
obj.GetAccountDetails()
obj.DisplayAccountDetails()
```

```
3266521
Keval
keval@gmail.com
saving
50000
```

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

In [6]:
```python
import math
class Circle:
    def area(self,r):
        print(f"Area:{math.pi*r*r}")

    def perimeter(self,r):
        print(f"Perimeter:{2*math.pi*r}")

obj=Circle()
obj.area(7)
obj.perimeter(7)
```

```
Area:153.93804002589985
Perimeter:43.982297150257104
```

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

In [14]:
```python
class Employee:
    def __init__(self,name,age,salary):
        self.name=name
        self.age=age
        self.salary=salary

    def updatedetails(self,name,age,salary):
        if name:
            self.name=name
        if age:
            self.age=age
        if salary:
            self.salary=salary

    def displaydetails(self):
        print("Employee Information:")
        print(f"Name: {self.name}")
        print(f"Age: {self.age}")
        print(f"Salary: {self.salary}")

obj=Employee("Keval",20,550000)
obj.displaydetails()
obj.updatedetails("Keval",19,2000000)
obj.displaydetails()
```

```
Employee Information:
Name: Keval
Age: 20
Salary: 550000
Employee Information:
Name: Keval
Age: 19
Salary: 2000000
```

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

In [18]:
```python
class Bank_Account:
    def __init__(self,acc_no,balance):
        self.acc_no=acc_no
        self.balance=balance

    def deposit(self,amount):
        if amount > 0:
            self.balance += amount
            print(f"Deposited Rs.{amount}. Current balance: Rs.{self.balance}")
        else:
            print("Deposit amount must be positive.")

    def withdraw(self,amount):
        if amount > 0:
            if amount <= self.balance:
                self.balance -= amount
                print(f"Withdrew Rs.{amount}. Current balance: Rs.{self.balance}")
            else:
```

```
                print("Insufficient funds.")
        else:
            print("Withdrawal amount must be positive.")

    def check_balance(self):
        print(f"Balance:Rs.{self.balance}")

obj=Bank_Account(151322,3000000)
obj.deposit(1500)
obj.withdraw(200)
obj.check_balance()
```

```
Deposited Rs.1500. Current balance: Rs.3001500
Withdrew Rs.200. Current balance: Rs.3001300
Balance:Rs.3001300
```

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 [20]:

```
class Inventory:
    def __init__(self):
        self.items = {}

    def add_item(self, item_name, price, quantity):
        if item_name in self.items:
            self.items[item_name]['quantity'] += quantity
        else:
            self.items[item_name] = {'price': price, 'quantity': quantity}
        print(f"Added {quantity} {item_name}(s) to inventory.")

    def remove_item(self, item_name, quantity):
        if item_name in self.items:
            if self.items[item_name]['quantity'] >= quantity:
                self.items[item_name]['quantity'] -= quantity
                print(f"Removed {quantity} {item_name}(s) from inventory.")
            else:
                print(f"Not enough {item_name} in inventory to remove.")
        else:
            print(f"{item_name} not found in inventory.")

    def update_price(self, item_name, new_price):
        if item_name in self.items:
            self.items[item_name]['price'] = new_price
            print(f"Updated price of {item_name} to ${new_price}.")
        else:
            print(f"{item_name} not found in inventory.")

    def display_inventory(self):
        if not self.items:
            print("Inventory is empty.")
        else:
            print("Inventory Details:")
            for item_name, details in self.items.items():
                print(f"{item_name}: Price - ${details['price']}, Quantity - {details['quantity']}")

inventory = Inventory()
inventory.add_item("Laptop", 1300, 20)
inventory.add_item("Phone", 900, 10)
inventory.display_inventory()
inventory.remove_item("Laptop", 50)
inventory.update_price("Phone", 950)
inventory.display_inventory()
inventory.remove_item("Laptop", 9)
inventory.update_price("Tablet", 3000)
```

```
Added 20 Laptop(s) to inventory.
Added 10 Phone(s) to inventory.
Inventory Details:
Laptop: Price - $1300, Quantity - 20
Phone: Price - $900, Quantity - 10
Not enough Laptop in inventory to remove.
Updated price of Phone to $950.
Inventory Details:
Laptop: Price - $1300, Quantity - 20
Phone: Price - $950, Quantity - 10
Removed 9 Laptop(s) from inventory.
Tablet not found in inventory.
```

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

In [22]:

```
class Car:
    def __init__(self, make, model, year, color):
```

```
        self.make = make
        self.model = model
        self.year = year
        self.color = color

    def display_car_info(self):
        print(f"Car Information:")
        print(f"Make: {self.make}")
        print(f"Model: {self.model}")
        print(f"Year: {self.year}")
        print(f"Color: {self.color}")

car1 = Car("Toyota", "Century", 2021, "Blue")
car1.display_car_info()
car2 = Car("Honda", "Civic", 2020, "Red")
car2.display_car_info()
```

```
Car Information:
Make: Toyota
Model: Century
Year: 2021
Color: Blue
Car Information:
Make: Honda
Model: Civic
Year: 2020
Color: Red
```

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 [26]:
```
class StudentKit:
    principal = "Mr ABC"

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

    def record_attendance(self, days_present):
        self.attendance = days_present
        print(f"Attendance recorded: {self.attendance} days")

    def generate_certificate(self):
        if self.attendance >= 75:
            print(f"Certificate of Attendance\n")
            print(f"Principal: {StudentKit.principal}")
            print(f"Student: {self.student_name}")
            print(f"Days Present: {self.attendance}")
            print(f"Status: Passed (Attendance is sufficient)")
        else:
            print(f"Certificate of Attendance\n")
            print(f"Principal: {StudentKit.principal}")
            print(f"Student: {self.student_name}")
            print(f"Days Present: {self.attendance}")
            print(f"Status: Failed (Attendance is insufficient)")

student_name = input("Enter the student's name: ")
student = StudentKit(student_name)
days_present = int(input("Enter the number of days present in class: "))
student.record_attendance(days_present)
student.generate_certificate()
```

```
Attendance recorded: 250 days
Certificate of Attendance

Principal: Mr ABC
Student: keval
Days Present: 250
Status: Passed (Attendance is sufficient)
```

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

In [28]:
```
class Time:
```

```python
    def __init__(self, hour=0, minute=0):
        self.hour = hour
        self.minute = minute

    def add_time(self, other):
        total_minutes = (self.hour * 60 + self.minute) + (other.hour * 60 + other.minute)
        total_hour = total_minutes // 60
        total_minute = total_minutes % 60
        return Time(total_hour, total_minute)

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

time1 = Time(2, 45)
time2 = Time(3, 30)
total_time = time1.add_time(time2)
print("Time 1:")
time1.display_time()
print("Time 2:")
time2.display_time()
print("Total Time after adding:")
total_time.display_time()
```

```
Time 1:
2 hour(s) and 45 minute(s)
Time 2:
3 hour(s) and 30 minute(s)
Total Time after adding:
6 hour(s) and 15 minute(s)
```

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 13

Keval Dhandhukiya

23010101064

Continued..

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

In [3]:
```
class Calculator:
    def __init__(self, length, width):
        self.length = length
        self.width = width

    def findArea(self):
        area = self.length * self.width
        print(area)

c = Calculator(20,30)
c.findArea()
```

```
600
```

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 CalcArea:
    area = 0
    def __init__(self, length):
        self.length = length

    def area(self):
        area = self.length * self.length
        self.output(area)

    def output(self,a):
        print(f"Output of Squre is {a}")

c = CalcArea(20)
c.area()
```

```
Output of Squre is 400
```

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 [7]:
```
class Calulatearea:
    length = 0
    width = 0
    def __init__(self,length, width):
        if length == width:
            print("Given value is about squre")
            return
        self.length = length
        self.width = width



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

    def output(self,a):
        print(f"Area of rect is {a}")


c = Calulatearea(20,40)
c.area()
```

```
Area of rect is 800
```

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 [11]:
```
class Simpleclass:
    __side = 0

    def __init__(self,side):
        self.__side = side

    def get_side(self):
        print(self.__side)

    def set_side(self, newSide):
        self.__side = newSide
        print(self.__side)

s = Simpleclass(30)
s.get_side()
s.set_side(50)
```

```
30
50
```

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 [13]:
```
class Profit:
    def get_profit(self, profit):
        self.profit = profit

class Loss:
    def get_loss(self,loss):
        self.loss = loss

class BalanceSheet(Profit, Loss):
    def getBalance(self):
        balance = self.profit - self.loss
        return balance

    def printBalance(self):
        print(self.getBalance())

blc = BalanceSheet()
blc.get_profit(3000)
blc.get_loss(400)
blc.printBalance()
```

```
2600
```

15) WAP to demonstrate all types of inheritance.

In [15]:
```
# Python program to demonstrate
# hybrid inheritance


class School:
    def func1(self):
        print("This function is in school.")


class Student1(School):
    def func2(self):
        print("This function is in student 1. ")


class Student2(School):
    def func3(self):
        print("This function is in student 2.")


class Student3(Student1, School):
    def func4(self):
        print("This function is in student 3.")


# Driver's code
object = Student3()
object.func1()
object.func2()


# ---------------------------------------------------------
# Python program to demonstrate
# Hierarchical inheritance


# Base class
class Parent:
    def func1(self):
        print("This function is in parent class.")

# Derived class1


class Child1(Parent):
    def func2(self):
        print("This function is in child 1.")

# Derivied class2


class Child2(Parent):
    def func3(self):
        print("This function is in child 2.")


# Driver's code
object1 = Child1()
object2 = Child2()
object1.func1()
object1.func2()
object2.func1()
object2.func3()

class Grandfather:

    def __init__(self, grandfathername):
        self.grandfathername = grandfathername


# ---------------------------------------------------------------------


class Father(Grandfather):
    def __init__(self, fathername, grandfathername):
        self.fathername = fathername

        # invoking constructor of Grandfather class
        Grandfather.__init__(self, grandfathername)

# Derived class
```

```
class Son(Father):
    def __init__(self, sonname, fathername, grandfathername):
        self.sonname = sonname

        # invoking constructor of Father class
        Father.__init__(self, fathername, grandfathername)

    def print_name(self):
        print('Grandfather name :', self.grandfathername)
        print("Father name :", self.fathername)
        print("Son name :", self.sonname)


#  Driver code
s1 = Son('Raj', 'Darshan', 'Chirag')
print(s1.grandfathername)
s1.print_name()
```

```
This function is in school.
This function is in student 1.
This function is in parent class.
This function is in child 1.
This function is in parent class.
This function is in child 2.
Chirag
Grandfather name : Chirag
Father name : Darshan
Son name : Raj
```

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 [23]:
```
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
```

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 [21]:
```
class Shape:
    def draw(self):
        return

class Rect(Shape):
    def draw(self):
        print("Rect")

class Cir(Shape):
    def draw(self):
        print("Cir")

class Tri(Shape):
    def draw(self):
        print("Tri")

rect = Rect()
cir = Cir()
tri = Tri()

li = [rect, cir, tri]

for i in li:
    i.draw()
```

```
Rect
Cir
Tri
```