Darshan UNIVERSITY

योगः कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 1

Drashti Ruparelia

23010101231

01) WAP to print “Hello World”

In [1]:
```
[print("Hello world")]
#shortform to run
#shift +enter
```

```
Hello world
```

Out[1]: [None]

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

In [7]:
```
print("addition without input")
a=5
b=3
# c=a+b
print("Addition of two number",a+b)
print("addition with input")
a=int(input("enter an number "))
b=int(input("enter another number "))
c=a+b
print("Addition of your two number is",c)
# andar tamare variable declare karvo hoi to f thi start apvani like print(f"add
```

```
addition without input
Addition of two number 8
addition with input
enter an number 12
enter another number 24
Addition of your two number is 36
```

03) WAP to check the type of the variable.

In [9]:
```
a=input("enter a number")
print(type(a))
# input by default string le che
```

```
enter a number45
<class 'str'>
```

04) WAP to calculate simple interest.

In [12]:
```
p=int(input("enter the principle "))
r=int(input("enter the rate "))
t=int(input("enter the time "))
print(f"simple interest of \n p:{p}\n r:{r}\n t:{t} \nis:",(p*r*t)/100)
```

```
enter the principle 2
enter the rate 3
enter the time 4
simple interest of
 p:2
 r:3
 t:4 is: 0.24
```

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

In [17]:
```
r=float(input("enter the radius "))
pie=3.14
print(f"Area of circle for radius {r} is",pie*r*r)
print(f"Perimeter of the circle for radius {r} is",2*pie*r)
```

```
enter the radius 3
Area of circle for radius 3.0 is 28.259999999999998
Perimeter of the circle for radius 3.0 is 18.84
```

06) WAP to calculate area of a triangle.

In [18]:
```
h=int(input("enter the height "))
b=int(input("enter the base "))
print(f"Area of triangle for height {h} base {b} is",1/2*b*h)
```

```
enter the height 2
enter the base 3
Area of triangle for height 2 base 3 is 3.0
```

07) WAP to compute quotient and remainder.

In [1]:
```
h=int(input("enter the divisior "))
b=int(input("enter the dividend "))
print("the quotient is ",b/h)
print("the remainder is ",b%h)
```

```
the quotient is  3.0
the remainder is  0
```

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

In [11]:
```
temp=int(input("enter the temperature "))
unit=input("enter the temp unit ")
if unit=="c":
    print("the temperature in fahrenhit is: ", (temp*9)/5+32)
else:
    print("the temperature in celsius is: ", (temp-32)*5/9)
```

```
the temperature in fahrenhit is:  33.8
```

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

In [9]:
```
import math
print("enter the x coordinate of both the points")
x1=int(input("enter the x coordinate of the first point"))
x2=int(input("enter the x coordinate of the second point"))
y1=int(input("enter the y coordinate of the first point"))
y2=int(input("enter the y coordinate of the second point"))
distance =math.sqrt((x2-x1)**2+(y2-y1)**2)
print(f"the distance between the {x1,y1} and {x2,y2} is",distance)
```

```
enter the x coordinate of both the points
the distance between the (2, 4) and (2, 4) is 0.0
```

10) WAP to print sum of n natural numbers.

In [19]:
```
n=int(input("enter the number "))
#print("answer is : ",n*(n+1)/2) or
sum=0
for i in range (n+1):
    sum+=i;
print("thus the answer is ",sum)
```

```
thus the answer is  6
```

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

In [25]:
```
n=int(input("enter the number "))
print("answer is: ",n*(n+1)*(2*n+1)/6)
```

```
answer is:  5.0
```

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

In [31]:
```
firstname=input("enter the first name of student")
lastname=input("enter the last name of student")
print("the full name of student: ",firstname,lastname)
```

```
the full name of student:  Drashti Ruparelia
```

13) WAP to swap two numbers.

In [37]:
```
num1=int(input("enter the number "))
num2=int(input("enter the number "))
print(f"before swap:\nnumber1: {num1}\nnumber2: {num2}")
temp=num1
```

```
num1=num2
num2=temp
print(f"after swap: \nnumber1: {num1}\nnumber2: {num2}")
```

```
before swap:
number1: 2
number2: 4
after swap:
number1: 4
number2: 2
```

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

In [39]:
```
distance=int(input("enter the distance in km "))
convertTo=input("enter the unit in which you want to convert the distance units
if convertTo=="m":
    print("the answer is: ",distance*1000)
elif convertTo=="ft":
    print("the answer is: ",distance*3280)
elif convertTo=="inch":
    print("the answer is: ",distance*39370)
else:
    print("the answer is: ",distance*100000)
```

```
the answer is:  200000
```

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

In [43]:
```
day=int(input("enter the day in km "))
month=int(input("enter the month in km "))
year=int(input("enter the year in km "))
print(f"thus the date is: {day}-{month}-{year}")
```

```
thus the date is: 23-8-2024
```

Python Programming - 2301CS404

Lab - 2

Drashti Ruparelia

23010101231

if..else..

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

In [1]:
```
num= int(input("enter the number "))
if num<0:
    print("number is negative")
else :
    print("number is positive")
```

```
enter the number -3
number is negative
```

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

In [2]:
```
num= int(input("enter the number "))
if num%2==0:
    print("number is even")
else :
    print("number is odd")
```

```
enter the number 3
number is odd
```

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

In [4]:
```
num1= int(input("enter the first number "))
num2= int(input("enter the second number"))
print("number",num1," is greater")if num1>num2 else print("number",num2," is gre
```

```
enter the first number 2
enter the second number3
number 3  is greater
```

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

In [5]:
```
num1= int(input("enter the first number "))
num2= int(input("enter the second number"))
num3= int(input("enter the third number"))
if num1>num2:

    if num1>num3:
        print("number",num1," is greater")
    else:
        print("number",num3," is greater")
else:
    if num2>num3:
        print("number",num2," is greater")
    else:
        print("number",num3," is greater")
```

```
enter the first number 1
enter the second number2
enter the third number3
number 3  is greater
```

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 [14]:
```
year= int(input("enter the year "))
if year%4==0 and year%100!=0  or  year%400==0:
    print(year," is a leap year")
else :
    print(year," is not a leap year")
```

```
enter the year 2000
2000  is a leap year
```

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

In [15]:
```
num1= int(input("enter the number "))
match num1:
    case 1:
        print(num1," is a sunday")
    case 2:
        print(num1," is a monday")
    case 3:
        print(num1," is a tuesday")
```

```
    case 4:
        print(num1," is a wednesday")
    case 5:
        print(num1," is a thursday")
    case 6:
        print(num1," is a friday")
    case 7:
        print(num1," is a saturday")
    case _:
        print("That's not a valid day of the week.")
```

```
enter the number 78
That's not a valid day of the week.
```

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

In [18]:
```
num1=int(input("enter the first number "))
num2=int(input("enter the second number "))
signs=input("enter the sign which function you need to operate ")
match signs:
    case "+":
        print("Ans is",num1+num2)
    case "-":
       print("Ans is",num1-num2)
    case "*":
        print("Ans is",num1*num2)
    case "/":
        print("Ans is",num1/num2)
```

```
enter the first number 4
enter the second number 2
enter the sign which function you need to operate-
Ans is 2
```

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 [23]:
```
num1=int(input("enter the marks in subj-1 "))
num2=int(input("enter the marks in subj-2 "))
num3=int(input("enter the marks in subj-3 "))
num4=int(input("enter the marks in subj-4 "))
num5=int(input("enter the marks in subj-5 "))
sum=num1+num2+num3+num4+num5
percentage = (sum/5)
print("your percentage is ",percentage);
if percentage<35:
    print("thus you are fail")
elif percentage<45:
    print("thus you are having a pass class")
elif percentage<60:
    print("thus you are having second class")
elif percentage<70:
    print("thus you are having first class")
```

```
else :
    print("thus you are having distinction")
```

```
enter the marks in subj-1 98
enter the marks in subj-2 99
enter the marks in subj-3 91
enter the marks in subj-4 95
enter the marks in subj-5 96
your percentage is  95.8
thus you are having distinction
```

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 [28]:
```
side1= int(input("enter the first side "))
side2= int(input("enter the second side "))
side3= int(input("enter the third side "))
if side1==side2 or side1==side3 or side1==side3:
    print("thus the given triangle is isosceles")
elif side1==side2==side3:
    print("thus the given triangle is equilateral")
elif side1**2==side2**2+side3**2 or side2**2==side1**2+side3**2 or side3**2==sid
    print("thus the given triangle is right angled triangle")
else:
    print("thus the triangle is scalene")
```

```
enter the first side 3
enter the second side 3
enter the third side 9
thus the given triangle is isosceles
```

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

In [7]:
```
num1= int(input("enter the first number "))
num2= int(input("enter the second number"))
num3= int(input("enter the third number"))
if num1>num2:
    if num1>num3:
             print("number",num3," is second Largest") if num3>num2 else print("
    else:
        print("number",num1,"is second largest")
elif num2>num3:
    if num1>num3:
            print("number",num1," is second Largest")
    else:
        print("number",num3," is second Largest")
else:
    if num3>num1:
        print("number",num2," is second largest")
    else:
        print("number",num1,"is second largest")
```

```
number 3  is second largest
```

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

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

In [9]:
```
units= int(input("enter your units "))
if units<=50:
    print("the bill is",units*50)
elif units<=100:
    print("the is bill is",(units*50+(units-50)*3.25))
elif units<=200:
    print("the is bill is",(units*50+(units-50)*3.25)+(units-100)*5.26)
else:
    print("the is bill is",(units*50+(units-50)*3.25)+(units-100)*5.26+(units-20
```

```
the bill is 2000
```

Python Programming - 2301CS404

Lab - 3

Drashti Ruparelia

23010101231

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 [5]:
```
n=int(input("enter the last number"))
for i in range(1,n+1):
    print(i)
```

```
1
2
3
4
5
```

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

In [15]:
```
n=int(input("enter the last number"))
for i in range(1,n+1):
    if i%2!=0:
        print(i)
```

```
1
3
5
7
9
11
13
15
17
19
21
```

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

In [19]:
```
n1=int(input("enter the first number"))
n2=int(input("enter the last number"))
for i in range(n1,n2):
    if i%2==0 and i%3!=0:
        print(i)
```

```
2
4
8
10
14
16
20
22
```

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

In [23]:
```
n=int(input("enter the last number"))
sum=0
for i in range(1,n+1):
     sum+=i
print("the answer is:",sum)
```

```
the answer is: 15
```

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

In [ ]:
```
n=int(input("enter the last number"))
for i in range(1,n+1):
    sum+=(i*i)
        print(i)
```

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

In [39]:
```
n=int(input("enter the last number"))
sum=0
for i in range(1,n+1):
    if i%2==0:
        sum+=(-i)
    else:
        sum+=i
print("sum:",sum)
```

```
sum: 2
```

08) WAP to print multiplication table of given number.

In [43]:
```
n=int(input("enter the number"))
for i in range(1,11):
    print(n,"*",i,"=",n*i)
```

```
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
3 * 4 = 12
3 * 5 = 15
3 * 6 = 18
3 * 7 = 21
3 * 8 = 24
3 * 9 = 27
3 * 10 = 30
```

09) WAP to find factorial of the given number.

In [51]:
```
n=int(input("enter the number"))
ans=1
for i in range(1,n+1):
    ans*=i
print("thus the answer is:",ans)
```

```
thus the answer is: 120
```

10) WAP to find factors of the given number.

In [53]:
```
n=int(input("enter the number to find the factor"))
print("factors of ",n,"is:")
for i in range(1,n+1):
    if n%i==0:
        print(i)
```

```
factors of  6 is:
1
2
3
6
```

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

In [59]:
```
n=int(input("enter the number"))
count=0
for i in range(1,n+1):
```

```
    if n%i==0:
        count+=1
if count>2:
    print("the number is not prime")
else:
    print("the number is prime")
```

```
number is prime
```

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

In [63]:
```
n=int(input("enter the number"))
sum=0
while n!=0:
    sum+=int(n)%10
    n=n/10
print("sum:",sum)
```

```
sum: 4
```

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

In [75]:
```
n=int(input("enter the number"))
temp=n
r=0
d=0
while temp!=0:
     r=temp%10
     temp=temp//10
     d=d*10+r
if d==n:
    print("the number is palindrome")
else:
    print("the number is not palindrome")
```

```
the number is palindrome
```

14) WAP to print GCD of given two numbers.

In [81]:
```
n1=int(input("enter the number to find the factor"))
n2=int(input("enter the another number to find the factor"))
gcd=0
for i in range(1,n1 if n1>n2 else n2):
    if n1%i==0 and n2%i==0:
        if gcd<i:
            gcd=i
print("greatest common factor is:",gcd)
```

```
greatest common factor is: 2
```

Python Programming - 2301CS404

Lab - 4

Drashti Ruparelia

23010101231

String

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

In [13]:
```
str=input("enter the String")
rev = str[::-1]
if y==str:
    print("string is palindrome")
else:
    print("string is not palindrome")
```

```
string is palindrome
```

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

In [19]:
```
str=input("enter the String")
str=str.split()
rev=str[-1:-len(str)-1:-1]
print(rev)
```

```
['girl', 'a', 'am', 'i']
```

03) WAP to remove ith character from given string.

In [27]:
```
str=input("enter the String")
index=int(input("enter the index"))
str1=str[0:index-1:1]
str2=str[index:len(str):1]
print("the string without that character "+(str1+str2))
```

```
the string without that character hell world
```

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

In [35]:
```
str=input("enter the String")
count=0;
for i in str:
   count=count+1
print("thus the length of the string is: ",count)
```

```
thus the length of the string is:  11
```

05) WAP to print even length word in string.

In [43]:
```
str=input("enter the String")
s1=str.split()
for i in s1:
    if len(i) %2==0:
        print(i)
```

```
am
girl
```

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

In [47]:
```
str=input("enter the String")
count=0
for i in str:
    if i=='a' or i=='u'or i=='i' or i=='o' or i=='e':
        count=count+1
print("count ",count)
```

```
count  3
```

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

In [67]:
```
str=input("enter the String")
s1=str.split()
s2=" "
for i in s1:
    s2=s2+" "+(i[0].upper()+i[1:len(i)-1]+i[-1].upper())
print(s2)
```

```
 HellO WorlD
```

08) WAP to convert given array to string.

In [57]:
```
li = ['hi', 'hello', 'how', 'are', 'you']
ans = " ".join(li)
print("thus the string is:",ans)
```

```
thus the string is: hi hello how are you
```

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

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

In [63]:
```
password=input("enter the passwords")
confirmpassword=input("enter the confirm passwords")
if password!=confirmpassword:
    print("make sure your passwords and confirm passwords matches each other")
```

```
make sure your passwords and confirm passwords matches each other
```

10) : Display credit card number.

card no. : 1234 5678 9012 3456

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

In [81]:
```
cardno=input("enter the card no")
s=cardno.split()
for i in s:
    if i[0:len(s)-5]:
        print("*")
else:
    print(i)
```

```
*
*
```

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

s1 = decimal and s2 = medical are Anagram

In [89]:
```
s1=input("enter the String")
s2=input("enter the String")
s=list(s1)
sn=list(s2)
s.sort()
sn.sort()
if s==sn:
    print("true")
```

```
true
```

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

input : EHIsarwiwhtwMV

output : IsarwiwhtwEHMV

In [8]:
```
input_str=input("enter the input string")
lower_case = ''.join([char for char in input_str if char.islower()])
upper_case = ''.join([char for char in input_str if char.isupper()])
print(lower_case + upper_case)
```

```
lsarwiwhtwEHMV
```

Python Programming - 2301CS404

Lab - 5

Drashti Ruparelia

23010101231

List

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

In [8]:
```
l1=[i for i in range(5)]
sum=0;
for i in l1:
    sum+=i
print("thus the sum is:",sum)
```

```
thus the sum is: 10
```

02) WAP to find largest element in a List.

In [18]:
```
n=int(input("enter the total number of element in list"))
l1=[]
for i in range (1,n+1):
  num=int(input("enter the number"))
  l1.append(num)
max=l1[1]
for i in l1:
    if max<i:
        max=i
print("the largest number is:",max)
```

```
the largest number is: 8
```

03) WAP to find the length of a List.

In [20]:
```
l1=[1,2,3]
print("thus the length of the list is",len(l1))
```

```
thus the length of the list is 3
```

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

In [30]:
```
n=int(input("enter the total number of element in list"))
l1=[]
for i in range (1,n+1):
  num=int(input("enter the number"))
  l1.append(num)

temp=l1[0]
l1[0]=l1[n-1]
l1[n-1]=temp
for i in l1:
    print(i)
```

```
3
2
1
```

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

In [60]:
```
l1=[5,8,9,7,4,5,9]
l2=l1[0:int(len(l1)/2)]
l3=l1[int(len(l1)/2):len(l1)]
l3.extend(l2)
print(l3)
```

```
[7, 4, 5, 9, 5, 8, 9]
```

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

In [34]:
```
n=int(input("enter the total number of element in list"))
l1=[]
for i in range (1,n+1):
  num=int(input("enter the number"))
  l1.append(num)
index1=int(input("enter the first index"))
index2=int(input("enter the second index"))
temp=l1[index1]
l1[index1]=l1[index2]
l1[index2]=temp
print(l1)
```

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

07) WAP to reverse the list entered by user.

In [36]:
```
n=int(input("enter the total number of element in list"))
l1=[]
for i in range (1,n+1):
  num=int(input("enter the number"))
  l1.append(num)
l1.reverse()
print(l1)
```

```
[3, 2, 1]
```

08) WAP to print even numbers in a list.

In [42]:
```
n=int(input("enter the total number of element in list"))
l1=[]
for i in range (1,n+1):
  num=int(input("enter the number"))
  l1.append(num)
print("thus the even numbers if present in list are as follows")
for i in l1:
    if i%2==0:
        print(i)
```

```
thus the even numbers if present in list are as follows
2
```

09) WAP to count unique items in a list.

In [68]:
```
n=int(input("enter the total number of element in list"))
l1=[]
for i in range (1,n+1):
  num=int(input("enter the number"))
  l1.append(num)
s=set(l1)
print("thus the unique numbers are as follows:",len(s))
print(s)
```

```
thus the unique numbers are as follows: 1
{3}
```

10) WAP to copy a list.

In [44]:
```
n=int(input("enter the total number of element in list"))
l1=[]
for i in range (1,n+1):
  num=int(input("enter the number"))
  l1.append(num)
l2=l1.copy()
print(l2)
```

```
[1, 2, 3]
```

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

In [48]:
```
n=int(input("enter the range of element in list"))
print("thus the odd number are as follows")
l1=[i for i in range(n+1) if i%2!=0]
print(l1)
```

```
thus the odd number are as follows
[1, 3, 5]
```

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

In [52]:
```
n=int(input("enter the total number of element in list"))
l1=[]
```

```
for i in range (1,n+1):
  num=int(input("enter the number"))
  l1.append(num)
count=0;
l1.sort()
num=int(input("enter the number of which you need the occurence"))
for i in l1:
    if i==num:
        count+=1
print("thus the occurence of",num,"is:",count)
```

```
thus the occurence of 3 is: 3
```

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

In [54]:
```
n=int(input("enter the total number of element in list"))
l1=[]
for i in range (1,n+1):
  num=int(input("enter the number"))
  l1.append(num)
l1.sort()
print("the second largest number is:",l1[n-2])
```

```
the second largest number is: 7
```

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

In [84]:
```
n=int(input("enter the total number of element in list"))
k=int(input("enter the frequency"))
l1=[]
for i in range (1,n+1):
  num=int(input("enter the number"))
  l1.append(num)
s=set(l1)
for i in s:
    num=l1.count(i)
    if num>k:
        print("thus the",num,"having frequency greater than",k)
```

```
thus the 2 having frequency greater than 1
```

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

In [94]:
```
l1= [i**2 for i in range(1,10)]
print(l1)

# n=int(input("enter the total number of element in list"))
# l1=[]
# for i in range (1,n+1):
#   num=int(input("enter the number"))
#   l1.append(num)
```

```
[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 [90]:
```
n=int(input("enter the total number of element in list"))
l1=[]
for i in range (1,n+1):
  fruit=input("enter the number")
  l1.append(fruit)
temp= [i for i in l1 if i[0] == 'b']
print(temp)
```

```
['bhgyugbh', 'bhygy']
```

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

In [7]:
```
n1=int(input("enter the total number of element in list1"))
l1=[]
for i in range (1,n1+1):
  num=int(input("enter the number"))
  l1.append(num)
n2=int(input("enter the total number of element in list2"))
l2=[]
for i in range (1,n2+1):
  num=int(input("enter the number"))
  l2.append(num)
l3=[]
l3=list(set(l1) & set(l2))
print("the common elements are: ",l3)
```

```
the common elements are:  [1, 2]
```

Python Programming - 2301CS404

Lab - 6

Drashti Ruparelia

23010101231

Tuple

01) WAP to find sum of tuple elements.

In [6]:
```
n = int(input("Enter total element of tuple"))
l=[]
for i in range(1,n+1):
    num = int(input("Enter element"))
    l.append(num)
t1 = tuple(l)
# -------or----------
# l = input("Enter elements separated by spaces: ").split()
# # Converting list to tuple
# t1 = tuple(l)
# store elements as a string

Sum = sum(t1)
print("Sum of tuple elments are: ",Sum)
```

```
Sum of tuple elments are:  6
```

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

In [2]:
```
n = int(input("Enter total element of tuple"))
l=[]
for i in range(1,n+1):
    num = int(input("Enter element"))
    l.append(num)
t1 = tuple(l)
k = int(input("Enter number that how many max and min elements you want to find
```

```
sortedCol = sorted(list(t1))
vals = tuple(sortedCol[:k] + sortedCol[-k:])
print("Tuple : ",t1)
print(f"{k} maximam and minimum values : {vals}")
```

```
Tuple :  (2, 5, 8, 1, 9, 4, 6)
2 maximam and minimum values : (1, 2, 8, 9)
```

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

In [4]:
```
k = int(input("Enter a number by which all the elements of the tuple are divisib
n = int(input("Enter total element of tuple"))
l=[]
for i in range(1,n+1):
    num = int(input("Enter element"))
    if num % k == 0:
        l.append(num)
t1 = tuple(l)
print(t1)
```

```
(5, 10, 15)
```

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

In [83]:
```
n = int(input("Enter total element of tuple"))
l=[]
for i in range(1,n+1):
    num = int(input("Enter element"))
    l.append(num)
Ans=[(i,pow(i,3)) for i in l]
print(Ans)
```

```
[(1, 1), (2, 8), (3, 27)]
```

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

In [85]:
```
n = int(input("Enter total element of tuple"))
l=[]
for i in range(1,n+1):
   x=input("enter the tuples space separated").split()#tuple enter karava mate a
   t=tuple(int(i) for i in x)
   l.append(t)
for i in l:
    if all(x>0 for x in i):
        print(i)
```

```
(1, 1, 1)
```

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

In [75]:
```
l=[]
#tuple to list
# for i in range(1,n+1):
x=input("enter the tuples space separated").split()#tuple enter karava mate aa b
```

```
t=tuple(int(i) for i in x)
l.append(t)#list to tuple mate ane comment kari nakhvani or else ee akhu list sa
# print(l)


#t1=tuple(l)
n = int(input("Enter total element of list"))

for i in range(1,n+1):
    num = int(input("Enter element"))
    l.append(num)
#     t1=tuple(l1)
print(l)

#list to tuple
l1=list(t)
l1.append(l)
res=tuple(l1)
print(res)
```

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

07) WAP to remove tuples of length K.

In [49]:
```
n = int(input("Enter total element of tuple"))
l=[]
for i in range(1,n+1):
   x=input("enter the tuples space separated").split()#tuple enter karava mate a
   t=tuple(int(i) for i in x)
   l.append(t)

k = int(input("Enter total length of tuple to be removed"))
for i in l:
    if len(i)==k:
        l.remove(i)
print(l)
```

```
[(1, 2, 3)]
```

08) WAP to remove duplicates from tuple.

In [21]:
```
n = int(input("Enter total element of tuple"))
l=[]
for i in range(1,n+1):
    num = int(input("Enter element"))
    l.append(num)
s1=set(l)
t1=tuple(s1)
print(t1)
```

```
(1, 2)
```

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

In [19]:
```
n = int(input("Enter total element of tuple"))
l=[]
```

```
for i in range(1,n+1):
    num = int(input("Enter element"))
    l.append(num)
t1 = tuple(l)
l1=[]
num=1
for i in range(0,n-1):
    num=l[i]*l[i+1]
    l1.append(num)
t2=tuple(l1)
print(t2)
```

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

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

In [7]:
```
n = int(input("Enter total element of tuple"))
l=[]
for i in range(1,n+1):
    num = int(input("Enter element"))
    l.append(num)
t1 = tuple(l)
s1=set(t1)
if len(t1)==len(s1):
    print("the tuple is distinct")
else:
    print("the tuple is not distinct")
```

```
the tuple is distinct
```

Python Programming - 2301CS404

Lab - 7

Drashti Ruparelia

23010101231

Set & Dictionary

01) WAP to iterate over a set.

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

```
1
2
3
4
5
```

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

In [4]:
```
s={1,2,3,4,5}
l=list(s)
t=tuple(s)
print("list:",l)
print("tuple:",t)
```

```
list: [1, 2, 3, 4, 5]
tuple: (1, 2, 3, 4, 5)
```

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

In [8]:
```
s={7,2,9,4,5}
s1=max(s)
s2=min(s)
print("the max:",s1)
print("the min:",s2)
```

```
the max: 9
the min: 2
```

04) WAP to perform union of two sets.

In [10]:
```
s1={7,2,9,4,5}
s2={1,2,3,4,5}
s=s1|s2
print(s)
```

```
{1, 2, 3, 4, 5, 7, 9}
```

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

In [16]:
```
l1=[7,2,9,4,5]
s1=set(l1)
l2=[1,2,4,5,3]
s2=set(l2)
s=s1&s2
if s!=set():
    print(s)
else:
    print("no common element")
```

```
{2, 4, 5}
```

06) WAP to remove duplicates from list.

In [18]:
```
l=[1,2,2,3,3,4,5,5]
print("before removing the duplicates",l)
s=list(set(l))
print("after removing the duplicates the list are",s)
```

```
before removing the duplicates [1, 2, 2, 3, 3, 4, 5, 5]
after removing the duplicates the list are [1, 2, 3, 4, 5]
```

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

In [27]:
```
s1=input("enter the sentence").split()
s=set(s1)
print("the unique words are:",s)
```

```
the unique words are: {'world', 'everyone', 'hello'}
```

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

In [25]:
```
a={7,2,9,4,5}
b={1,2,4,5,3}
s=a&b
a=a-s
print("thus the after removing the common element the set is",a)
```

```
thus the after removing the common element the set is {9, 7}
```

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

In [45]:
```
#decimal and medical are anagrams and looped and poodle
s1=input("enter the word1")
s2=input("enter the word2")

dict1 = {}
for i in s1:
    if i in dict1:
        dict1[i] += 1
    else:
        dict1[i] = 1

dict2 = {}
for i in s2:
    if i in dict2:
        dict2[i] += 1
    else:
        dict2[i] = 1

if dict1==dict2:
    print("the two string are anagram")
else:
    print("the two string are not anagram")
```

```
the two string are anagram
```

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

In [39]:
```
l1=[7,2,9,4,5]
a=set(l1)
l2=[1,2,4,5,3]
b=set(l2)
l3=[2,12,13,14]
c=set(l3)
s=a&b&c
print(s)
```

```
{2}
```

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

In [51]:
```
s1=input("enter the sentence")
count=0
s={'a','e','i','o','u'}
for i in s1:
    if i in s:
        count+=1
print(count)
```

```
7
```

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

In [65]:
```
s1=input("enter the sentence")
count=0
l=list(s1)

for i in l:
    if i!="1" and i!="0":
        print("the string is not binary")
        break
else:
    print("the string is binary")
```

```
the string is binary
```

13) WAP to sort dictionary by key or value.

In [67]:
```
dict1 = {'d': 5, 'c': 8, 'e': 2}
myKeys = list(dict1.keys())
myKeys.sort()
sd = {i: dict1[i] for i in myKeys}
print(sd)
```

```
{'c': 8, 'd': 5, 'e': 2}
```

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

In [1]:
```
dic = {'a': 10, 'b': 20, 'c': 30}
print("Sum :", sum(dic.values()) )
Sum : 60
dic = {'a': 10, 'b': 20, 'c': 30}
def sum_dict_values(d):
     return sum(d.values())
sum_dict_values(dic)
```

```
Sum : 60
```

Out[1]:
```
60
```

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 [5]:
```
d1 = {'a': 5, 'c': 8, 'e': 2}
result = d1.get('d', "Key Not Found")
print(result)
#or
d1 = {'a': 5, 'c': 8, 'e': 2}
def handle_missing_key(d, key):
     return d.get(key, "Key Not Found")
handle_missing_key(d1, 'd')
```

```
Key Not Found
```

Out[5]:
```
'Key Not Found'
```

Python Programming - 2301CS404

Lab - 8

Drashti Ruparelia

23010101231

User Defined Function

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

In [5]:
```
def BMI(mass,height):
    return (mass/height**2)
m = int(input("Enter Mass: "))
h = int(input("Enter Height: "))
print("thus the BMI:",BMI(m,h))
```

```
thus the BMI: 0.4
```

02) Write a function that add first n numbers.

In [9]:
```
def Add(n):
    sum=0
    for i in range(1,n+1):
        sum+=i
    return sum
n = int(input("Enter last number: "))
print("Thus the Sum:",Add(n))
```

```
Thus the Sum: 6
```

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

In [11]:
```
def prime(n):
    count=0
    for i in range(1,n+1):
```

```
        if n%i==0:
            count+=1
    if count==2:
        return 1
    else:
        return 0
n = int(input("Enter the number: "))
prime(n)
```

Out[11]: 1

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

In [17]:
```
def prime(n,m):
    l1=[]
    for i in range(n+1,m):
        count=0
        for j in range(1,i+1):
            if i%j==0:
                count+=1

        if count==2:
            l1.append(i)
    return l1
n = int(input("Enter the number1: "))
m = int(input("Enter the number2: "))
print(f"the list of prime number between {n} to {m}:",prime(n,m))
```

```
the list of prime number between2 to10: [3, 5, 7]
```

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

In [31]:
```
def Palindrome(s1):
    rev=s1[::-1]
    if s1==rev:
        return True
    else:
        return False
s1 = input("Enter the string: ")
print(Palindrome(s1))
```

```
True
```

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

In [37]:
```
def Sum(l1):
    # sum=0
    # for i in l1:
    #     sum+=i
    # return sum
    #or
    return sum(l1)
n = int(input("Enter the total number in the list: "))
l1=[]
```

```
for i in range(1,n):
    n = int(input("Enter the  number in the list: "))
    l1.append(n)
print("Thus your list is:",l1)
print("Thus the Sum is:",sum(l1))
```

```
Thus your list is: [1, 2, 3]
Thus the Sum is: 6
```

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

In [51]:
```
def SumOfTuple(l1):
    s=0
    for i in l1:
        s+=i[0]
    return s
l1=[(1,2,3),(4,5,6),(2,3,5)]
print(l1)
print(SumOfTuple(l1))
```

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

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

In [67]:
```
def Fibo(n):
    if n==0 or n==1:
        return n
    else:
        return Fibo(n-2)+Fibo(n-1)
Fibo(4)
```

Out[67]: 3

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 [73]:
```
def name(d1,roll):
    return d1[roll]
d1={101:'Ajay', 102:'Rahul', 103:'Jay', 104:'Pooja'}
print(d1)
roll=int(input("enter roll number of student to find their name: "))
print(f"name of the student who has roll no{roll} is:{name(d1,roll)}")
```

```
{101: 'Ajay', 102: 'Rahul', 103: 'Jay', 104: 'Pooja'}
name of the student who has roll no101 is:Ajay
```

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

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

Ans = 200 + 300 + 100 = 600

In [75]:
```
def sumOfScore(score):
    sum=0
    for i in score:
        if i%10==0:
            sum+=i
    return sum
scores = [200, 456, 300, 100, 234, 678]
print("the sum is:",sumOfScore(scores))
```

```
the sum is: 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 [85]:
```
def invertdict(d1):
    l1=[]
    l2=[]
    for i in d1.keys():
        l1.append(i)
    for i in d2.keys():
        l2.append(d1[i])

Before = {'a': 10, 'b':20, 'c':30, 'd':40}
print(Before)
```

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

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 [2]:
```
def is_pangram(s):
     alphabet = "abcdefghijklmnopqrstuvwxyz"
     for i in alphabet:
         if i not in s.lower():
             return False
         return True
print(is_pangram("The quick brown fox jumps over the lazy dog"))
print(is_pangram("Hello World"))
```

```
True
False
```

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 [95]:
```
def Count(s1):
    upperCount=0
    lowerCount=0
    for i in s1:
        if i.isupper()==True:
            upperCount+=1
        else:
            lowerCount+=1
    print("uppercount:",upperCount,"LowerCount:",lowerCount)
s1 = input("Enter the string: ")
Count(s1)
```

```
uppercount: 3 LowerCount: 5
```

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

In [4]:
```
smaller = lambda a, b : a if a < b else b
print("Smaller of two : ",smaller(10, 20))
print("Smaller of two : ",smaller(5, 2))
```

```
Smaller of two :  10
Smaller of two :  2
```

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

In [6]:
```
names = ["Java", "DataStructure", "Flutter", "WebTechnology", "Dart",
"Python"]
long_names = list(filter(lambda name :len(name)>7, names))
print("Names of length>7 : ",long_names)
```

```
Names of length>7 :  ['DataStructure', 'WebTechnology']
```

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

In [10]:
```
names = ["firstName", "middleName", "lastName"]
capitalized_names = list(map(lambda name: name.capitalize(),names))
print(capitalized_names)
```

```
['Firstname', 'Middlename', 'Lastname']
```

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 [14]:
```
#1. Positional Arguments
def student_info(name, age):
```

```
     print("Name :", name)
     print("Age :", age)
student_info("Drashti", 20)

#2. Keyword Arguments
def student_info(name, age) :
     print("Name :", name)
     print("Age :", age)
student_info(age=20, name="Drashti")

#3. Default Arguments
def student_info(name, age=19) :
     print("Name :", name, ", Age :", age)

student_info("Ruparelia")
student_info("Drashti", 20)

#4. Variable-Length Positional Arguments (*args)
# jitne chahe utne args de sakte hai
def sum_numbers(*args) :
    return sum(args)

print(sum_numbers(1, 2, 3, 4, 5))
```

```
Name : Drashti
Age : 20
Name : Drashti
Age : 20
Name : Ruparelia , Age : 19
Name : Drashti , Age : 20
15
```

Python Programming - 2301CS404

Lab - 9

Drashti Ruparelia

23010101231

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 [22]:
```
fp=open("file1.txt","r")
# print(fp.read())
# print(fp.readline())
print(fp.readlines())
fp.close()
```

```
['DRASHTI RUPARELIA']
```

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

In [26]:
```
fp=open("new.txt","x")
fp.close()
```

```
---------------------------------------------------------------------------
FileExistsError                           Traceback (most recent call last)
Cell In[26], line 1
----> 1 fp=open("new.txt","x")
      2 fp.close()

File ~\python\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.

In [52]:
```
fp=open("file1.txt","r")
print([fp.readline().strip() for i in range(1,6)])
fp.close()
```

```
['DRASHTI RUPARELIA', 'dra', 'sht', 'i', 'rupare lia']
```

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

In [58]:
```
fp=open("file1.txt","r")
wl=fp.read().split()
print(wl)
w_len=list(map(len,wl))
print(w_len)
max_len=max(w_len)
ans=[i for i in wl if len(i)==max_len]
print(ans)
```

```
['DRASHTI', 'RUPARELIA', 'dra', 'sht', 'i', 'rupare', 'lia', 'drash', 'ti']
[7, 9, 3, 3, 1, 6, 3, 5, 2]
['RUPARELIA']
```

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

In [82]:
```
fp=open("file1.txt","r")
wl=fp.read().split()
print("number of words:",len(wl))
fp.seek(0,0)
line=fp.readlines()
print("number of lines:",len(line))
w_len=list(map(len,wl))
print("number of character:",sum(w_len))
fp.close()
```

```
number of words: 9
number of lines: 6
number of character: 39
```

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

In [86]:
```
fp=open("file1.txt","r")
l1=fp.readlines()
fp1=open("file2.txt","w")
fp1.writelines(l1)
fp1.close()
fp.close()
```

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

In [88]:
```
fp=open("file1.txt")
print("size of the text file:",fp.seek(0,2))
fp.close()
```

```
size of the text file: 47
```

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

In [ ]:
```
def frequency
word=input("enter the word")
wl=fp.read().split()
return wl.count(word)
fp=open("file")
print(fp)
```

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 [118…
```
marks=input("enter 5 subjects marks space separated:").split()
print("max score is:",max(marks))
marks=[i+'\n' for i in marks]
fp=open("file3.txt","w+")
fp.writelines(marks)
fp.seek(0,0)
print(fp.read())
fp.close()
```

```
max score is: 30
10
20
30
```

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

(Note: each number should be in new line)

In [114…
```
fp=open("primenumber.txt","w")
l1=[]
for i in range(1,100):
    count=0
    for j in range(1,i+1):
        if i%j==0:
          count+=1
    if count==2:
        l1.append(i)


l1=[str(i)+'\n' for i in l1]
print(l1)
fp.writelines(l1)
fp.close()
```

```
['2\n', '3\n', '5\n', '7\n', '11\n', '13\n', '17\n', '19\n', '23\n', '29\n', '31
\n', '37\n', '41\n', '43\n', '47\n', '53\n', '59\n', '61\n', '67\n', '71\n', '73
\n', '79\n', '83\n', '89\n', '97\n']
```

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

In [20]:
```
fp1=open("file3.txt","r+")

fp2=open("primenumber.txt","r+")
fp3=open("new1.txt","w")
data=fp1.read()+'\n'+fp2.read()

fp3.write(data)
fp1.close()
fp2.close()
fp3.close()
```

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

In [3]:
```
first="dra"
second="sht"
fp=open("file1.txt","r+")
data=fp.read()
data = data.replace(first,second)
print(data)
fp.close()
```

```
DRASHTI RUPARELIA
sht
sht
i
rupare lia
shtsh ti
```

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

In [64]:
```
fp=open("file1.txt","r")
print("file pointer initially is at: ",fp.tell())
fp.read(3)
print("after reading 3 character filepointer is at:",fp.tell())
fp.read()
print("the last pointer is at:",fp.tell())
print("the seeking pointer is at beginning:",fp.seek(1,0))
fp.read(3)
print("the seeking pointer is at current position:",fp.seek(0,1))
print("the seeking pointer is at end:",fp.seek(0,2))
fp.close()
```

```
file pointer initially is at:  0
after reading 3 character filepointer is at: 3
the last pointer is at: 47
the seeking pointer is at beginning: 1
the seeking pointer is at current position: 4
the seeking pointer is at end: 47
```

Python Programming - 2301CS404

Lab - 10

Drashti Ruparelia

23010101231

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 [3]:
```
try:
    a=int(input("enter the first number"))
    b=int(input("enter the second number"))
    #for value error remove int from starting because value error occures when o
    #other is string and there is a mathematical operand in betwen
    c=a//b
except ZeroDivisionError as msg:
    print(msg)
except ValueError as msg:
    print(msg)
except TypeError as msg:
    print(msg)
```

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

02) WAP to handle following exceptions:

1. IndexError
2. KeyError

In [17]:
```
#index error
l1=[1,2,3,4]
try:
    print(l1[4])
except IndexError as msg:
    print(type(msg).__name__+ ":" +str(msg))

#key error
ages = {'Jim': 30, 'Pam': 28, 'Kevin': 33}
try:
    print(ages['Micheal'])
except KeyError as msg:
    print(type(msg).__name__+ ":" +str(msg))
```

```
IndexError:list index out of range
KeyError:'Micheal'
```

03) WAP to handle following exceptions:

1. FileNotFoundError
2. ModuleNotFoundError

In [23]:
```
#filenotFound error
try:
    fp=open("abc.txt","r")
except Exception as msg:
    print(type(msg).__name__+ ":" +str(msg))
else:
    print(fp.read())
    fp.close()

#ModuleNotFound
try:
    import drashti
except ModuleNotFoundError:
    print("The module is not installed")
else:
    # Code to run if the module  is successfully imported
    print(" Module is installed")
```

```
FileNotFoundError:[Errno 2] No such file or directory: 'abc.txt'
The module is not installed
```

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

In [29]:
```
try:
    a=int(input("enter the first number"))
    b=int(input("enter the second number"))
    c=a//b
except Exception as msg:
    print(type(msg).__name__+ ":" +str(msg))
else:
    print(c)
```

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

05) WAP to demonstrate else and finally block.

In [33]:
```
try:
    a=int(input("enter the first number"))
    b=int(input("enter the second number"))
    c=a//b
except Exception as msg:
    print(type(msg).__name__+ ":" +str(msg))
else:
    #exception ma jase to else execute nai thai pan finally thase pan exception
    print(c)
    print("hello else")
finally:
    print("hello finally")
```

```
ZeroDivisionError:integer division or modulo by zero
hello finally
```

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 [73]:
```
l1=input("enter the list of grades separated by comma: ").split(',')
try:
    li=[int(i) for i in l1]
    print(li)
except Exception as msg:
    print(type(msg).__name__+ ":" +str(msg))
    print("cannot convert string to int")
```

```
ValueError:invalid literal for int() with base 10: 'a'
cannot convert string to int
```

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

In [65]:
```
def divide(a,b):
    return a//b
try:
    a=int(input("enter the first number"))
    b=int(input("enter the second number"))
    divide(a,b)
except ZeroDivisionError as msg:
    print(type(msg).__name__+ ":" +str(msg))
```

```
ZeroDivisionError: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 [67]:
```
try:
    age=int(input("enter the age:"))
    if age<18:
        raise ValueError("invalid age")
except ValueError:
    print("enter the valid age")
```

```
enter the valid age
```

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

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

otherwise print the given username.

In [69]:
```
class InvalidUsernameError(Exception):
    def __init__(self,msg):
        self.msg=msg
try:
    uname=input("enter the username")
    if len(uname)<5 or len(uname)>15:
        raise InvalidUsernameError("enter the valid username")
except InvalidUsernameError as msg:
  print(type(msg).__name__+ ":" +str(msg))
```

```
InvalidUsernameError:enter the vallid username
```

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 [71]:
```
class NegativeNumberError(Exception):
    def __init__(self,msg):
        self.msg=msg
try:
    num=int(input("enter the num:"))
    if num<0:
        raise NegativeNumberError("cannot calculate the square root of negative
    else:
        print(num**(1/2))
except NegativeNumberError as msg:
    print(type(msg).__name__+ ":" +str(msg))
```

```
NegativeNumberError:cannot calculate the square root of negative number
```

Darshan UNIVERSITY
योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 11

Drashti Ruparelia

23010101231

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 [7]:
```
import module1 as m
num1=int(input("enter the first number"))
num2=int(input("enter the second number"))
print(m.add(num1,num2))
print(m.sub(num1,num2))
print(m.multiply(num1,num2))
print(m.div(num1,num2))
```

```
15
5
50
2.0
```

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

In [19]:
```
import random
st=input("enter the first number")
index=random.randint(1,len(st))
print(st[index])
```

```
i
```

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

In [27]:
```
import random
list1=(23,8,200,5)
random.choice(list1)
```

Out[27]: 23

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

In [164…
```
import random
list1=(1,2,3,4,5,6)
random.seed(2)
print(random.choice(list1))

random.seed(1)
print(random.choice(list1))

random.seed(7)
print(random.choice(list1))
```

```
1
2
3
```

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

In [53]:
```
import random
# print(random.randrange(100,999,5))
l1=[]
for i in range(3):
    l1.append(random.randrange(100,999,5))
print(l1)
```

```
[315, 955, 490]
```

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 [128…
```
import random
l1=[]
l1=[i for i in range(1,1000)]
l2=random.sample(l1,100)
print(l2)
winner=random.choice(l2)
# # print(winner)
l2.remove(winner)
# # print(l2)
runner=random.choice(l2)
print("winner",winner)
print("runner",runner)
```

```
[161, 781, 958, 869, 156, 417, 930, 485, 591, 383, 916, 549, 76, 677, 805, 948, 4
41, 342, 945, 69, 628, 74, 470, 168, 864, 423, 245, 720, 561, 346, 934, 437, 855,
882, 966, 655, 543, 509, 661, 814, 800, 658, 252, 969, 355, 820, 681, 500, 773, 6
13, 14, 848, 281, 538, 899, 215, 49, 328, 715, 595, 306, 548, 251, 371, 51, 449,
292, 84, 376, 434, 669, 833, 736, 614, 183, 158, 106, 946, 220, 951, 241, 521, 94
1, 606, 226, 455, 310, 699, 557, 756, 853, 552, 353, 197, 812, 589, 867, 54, 179,
751]
winner 658
runner 736
```

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

In [106…

```
import datetime as d
print(d.datetime.today())
```

```
2025-02-13 13:08:16.037350
```

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

In [166…

```
import datetime
today=datetime.datetime.now()
df7=today-datetime.timedelta(days=7)
print(df7)
```

```
2025-02-06 13:25:49.455995
```

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

In [2]:

```
from datetime import datetime
import math
d1 = datetime(2025, 1, 1)
d2 = datetime(2025, 2, 10)
df = abs((d2 - d1).days)
print("Number of Days Between the Dates :", df)
```

```
Number of Days Between the Dates : 40
```

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

In [168…

```
import datetime

def find_day_of_week(date):
    day, month, year = (int(x) for x in date.split(' '))
    born = datetime.date(year, month, day)
    return born.strftime("%A")

# Input date in DD MM YYYY format
date = input("Enter date (DD MM YYYY): ")
print("The day of the week is:", find_day_of_week(date))
```

```
The day of the week is: Tuesday
```

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

In [4]:
```
import datetime
d1 = datetime.datetime.today()
dayy = d1.strftime("DATE : %d, %B %Y and TIME : %H : %M : %S")
print(dayy)
```

```
DATE : 13, March 2025 and TIME : 18 : 22 : 57
```

In [6]:
```
from datetime import datetime, timedelta
now = datetime.now()
print("Current Date and Time :", now)
formatted_date = now.strftime("%d-%m-%Y %H:%M:%S")
print("Formatted Date and Time :", formatted_date)
future_date = now + timedelta(days=5)
print("Date after 5 Days :", future_date.date())
past_time = now - timedelta(hours=3)
print("Time 3 Hours Ago :", past_time.time())
```

```
Current Date and Time : 2025-03-13 18:22:59.563916
Formatted Date and Time : 13-03-2025 18:22:59
Date after 5 Days : 2025-03-18
Time 3 Hours Ago : 15:22:59.563916
```

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

In [8]:
```
import math
print(f"Ceil: {math.ceil(5.2)}")
print(f"Floor: {math.floor(6.8)}")
print("Square Root of 25:", math.sqrt(25))
print("2 raised to the power 3:", math.pow(2, 3))
print("Factorial of 5:", math.factorial(5))
print("Sin of 90 degrees:", math.sin(math.radians(90)))
print("Cos of 0 degrees:", math.cos(math.radians(0)))
print("Value of Pi:", math.pi)
print("Value of Euler's number (e):", math.e)
```

```
Ceil: 6
Floor: 6
Square Root of 25: 5.0
2 raised to the power 3: 8.0
Factorial of 5: 120
Sin of 90 degrees: 1.0
Cos of 0 degrees: 1.0
Value of Pi: 3.141592653589793
Value of Euler's number (e): 2.718281828459045
```

Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 12

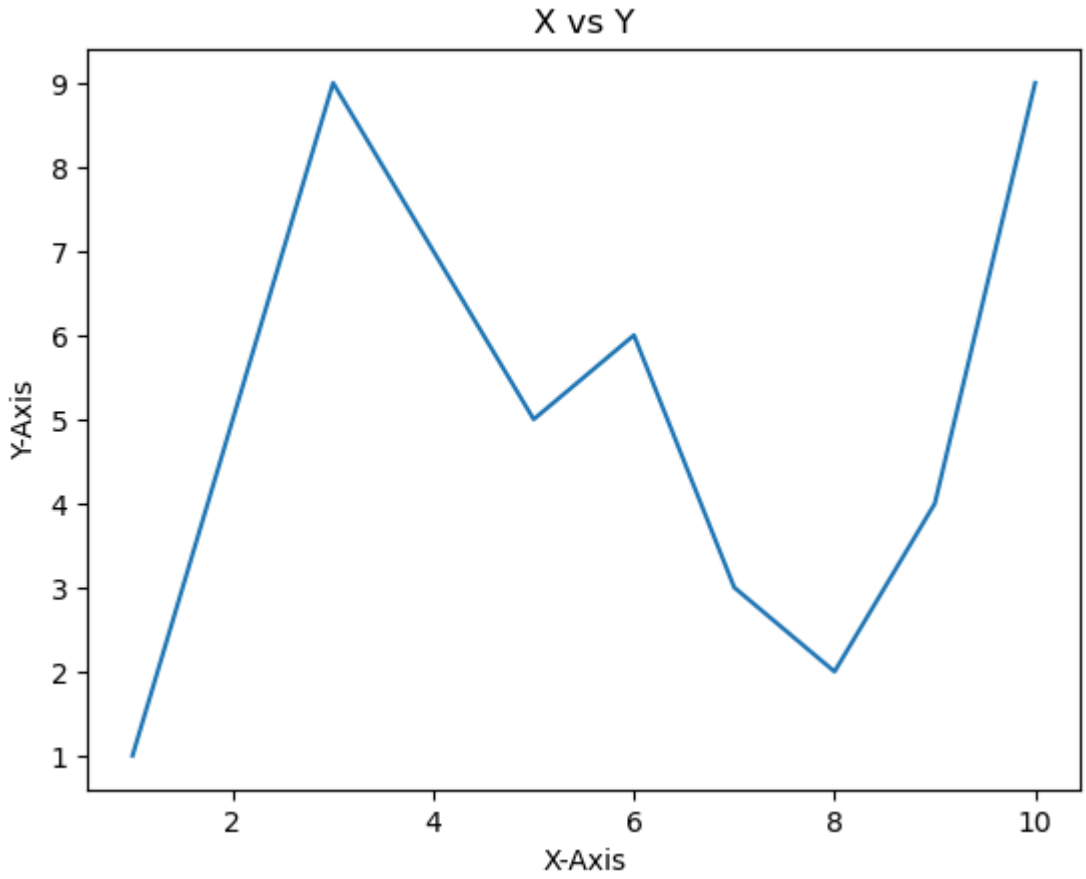
Drashti Ruparelia

23010101231

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

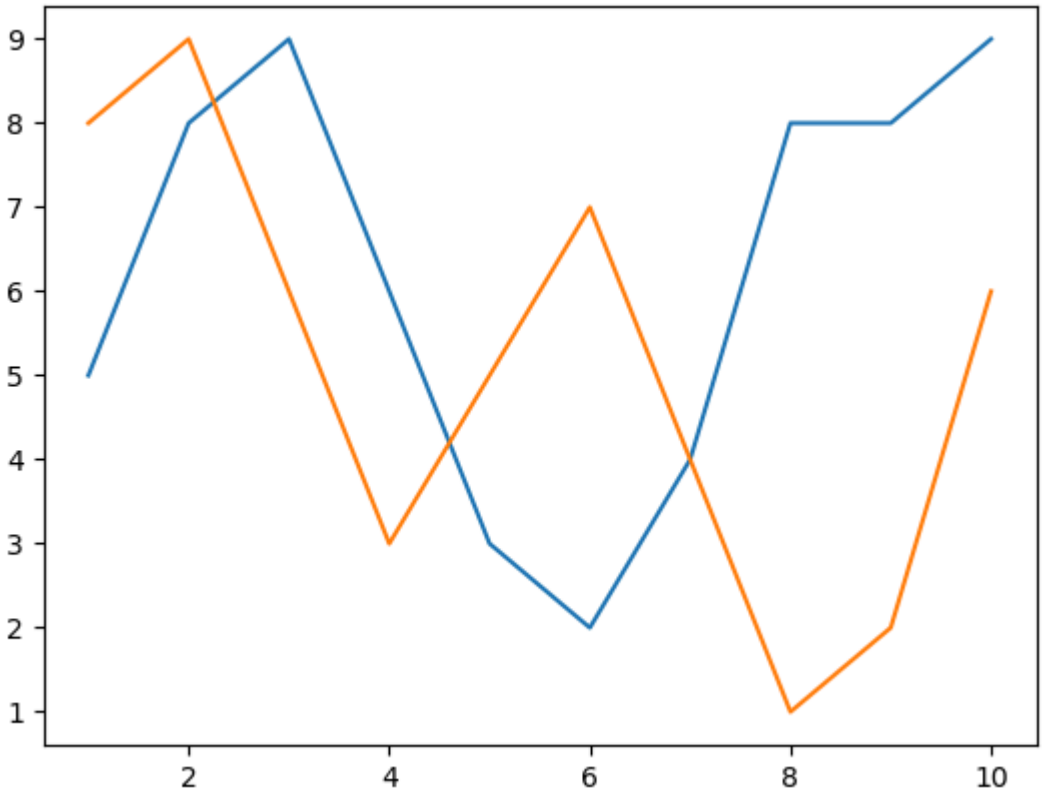
1. write a code to display the line chart of above x & y

In [9]:
```
x = range(1,11)
y = [1,5,9,7,5,6,3,2,4,9]
plt.plot(x,y)
plt.xlabel("X-Axis")
plt.ylabel("Y-Axis")
plt.title("X vs Y")
plt.show()
```



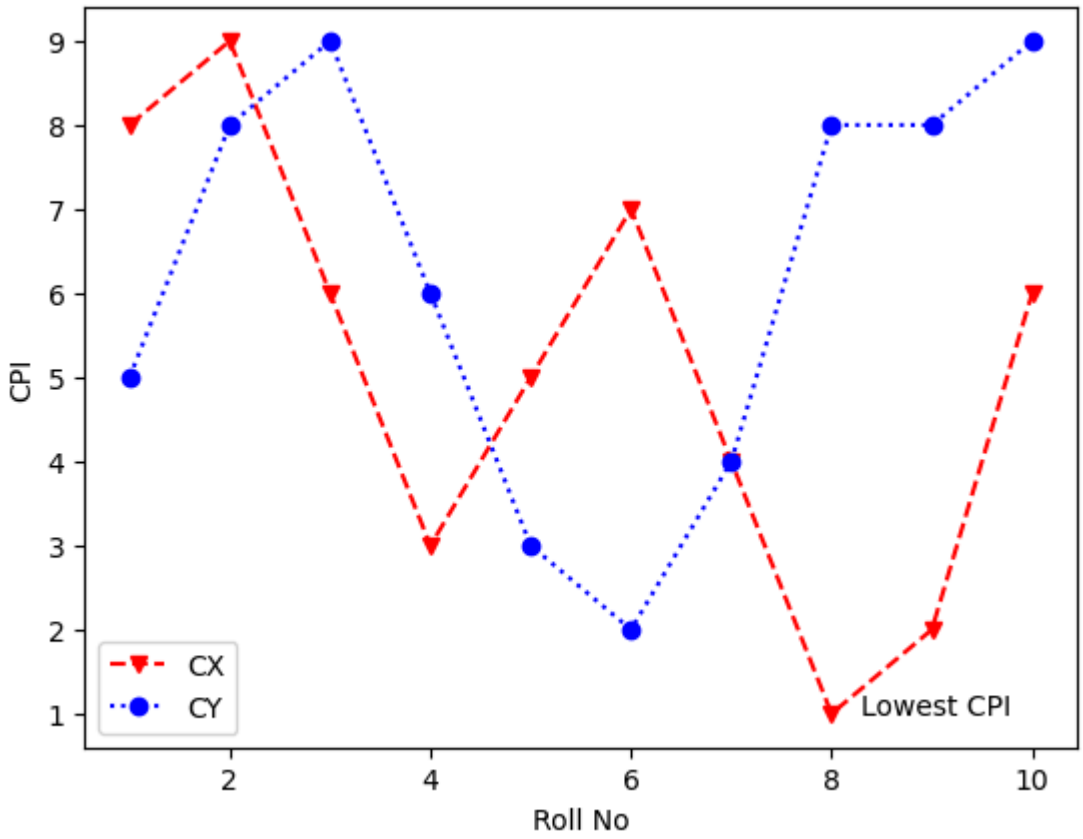
2. write a code to display two lines in a line chart (data given above)

In [14]:
```
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)
plt.plot(x,cyMarks)
plt.show()
```



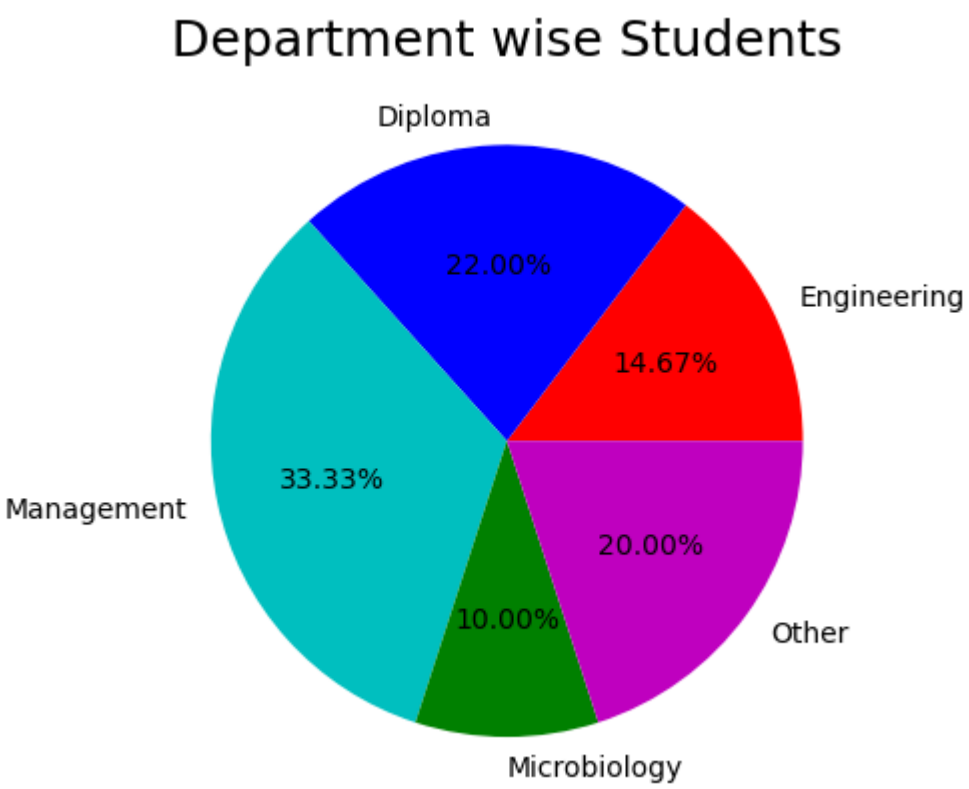
3. write a code to generate below graph

In [19]:
```
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]
#r-- means dashed red line, marker are for the pounts and label is for corner
plt.plot(x, cxMarks, 'r--', marker='v', label="CX")
# similarly b: is blue dotted line
plt.plot(x, cyMarks, 'b:', marker='o', label="CY")
plt.annotate("Lowest CPI", xy=[8.3,1])
plt.xlabel("Roll No")
plt.ylabel("CPI")
plt.legend()
plt.show()
```

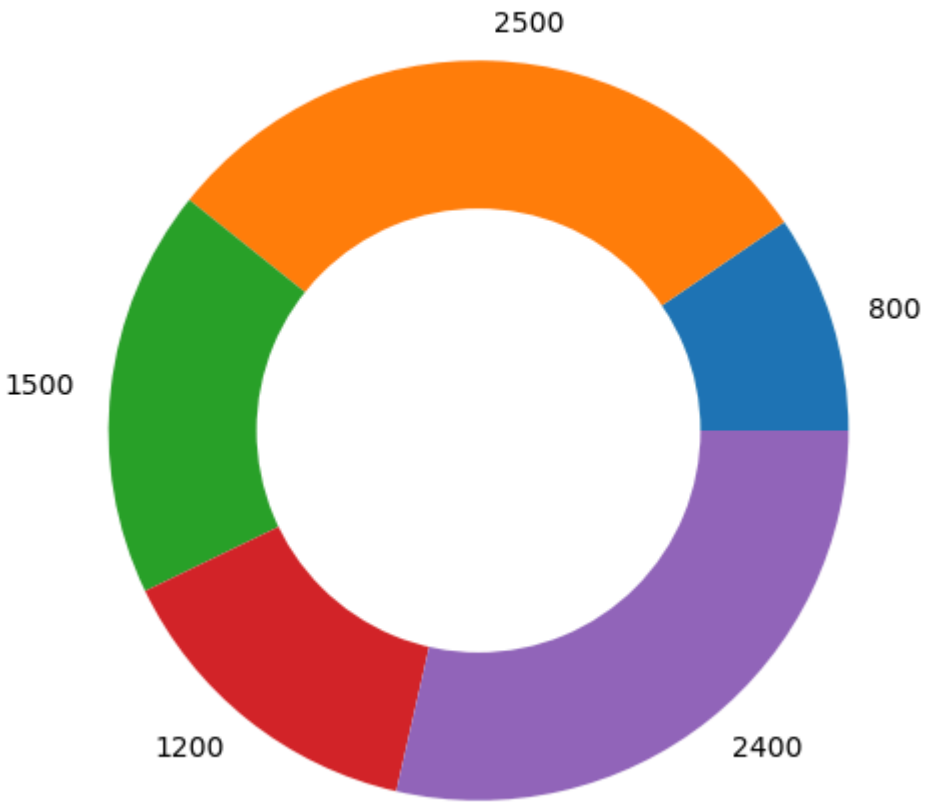


04) WAP to demonstrate the use of Pie chart.

In [37]:
```
dept = ["Engineering","Diploma","Management","Microbiology","Other"]
students = [220, 330, 500, 150, 300]
#x for colors
x = ['r','b','c','g','m']
#or x = ['#d32f2f', '#ffeb3b', 'g', '#ff9800','b']
plt.pie(students, labels= dept, radius=1,colors=x, autopct="%1.2f%%")
plt.title("Department wise Students", fontsize=18)
plt.show()
```

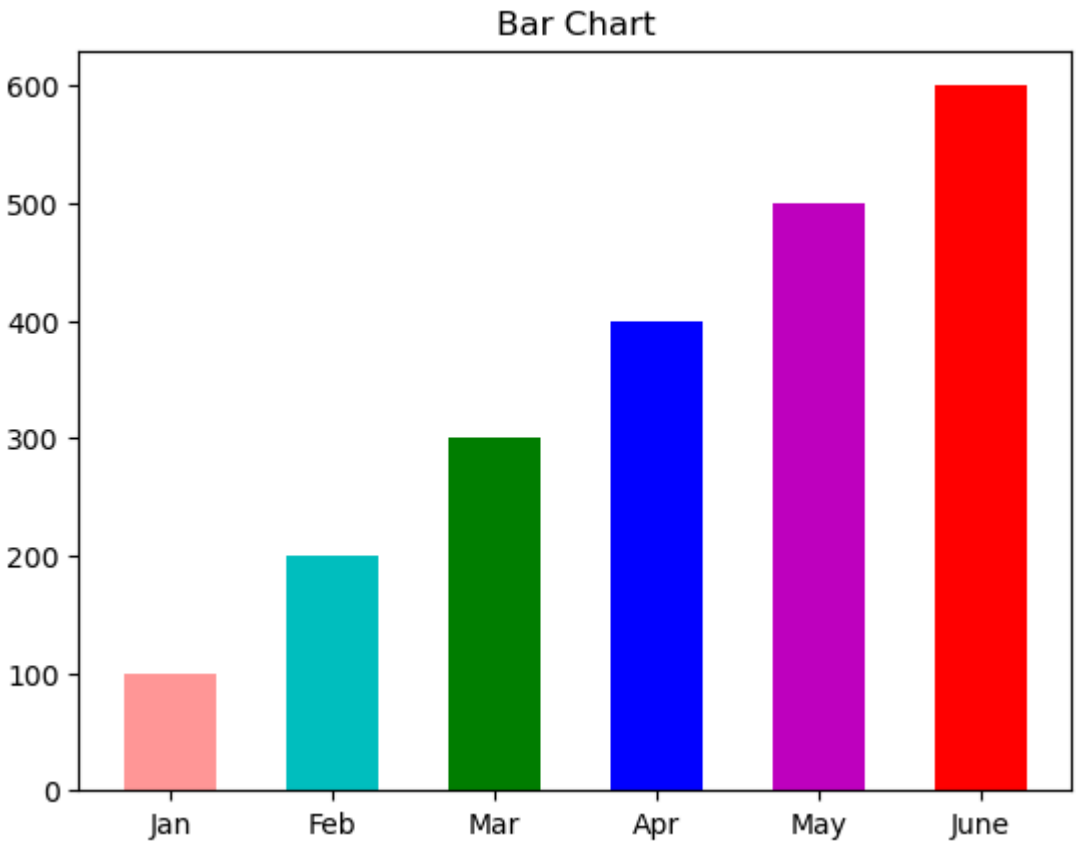


In [23]:
```
x1 = [800, 2500, 1500, 1200, 2400]
x2 = [1111]
plt.figure(figsize=(10,6))
plt.pie(x1, radius=1,labels=x1)
plt.pie(x2, radius=0.6, colors="w")
plt.title("Ring/Donut Chart",fontsize=20)
plt.show()
```

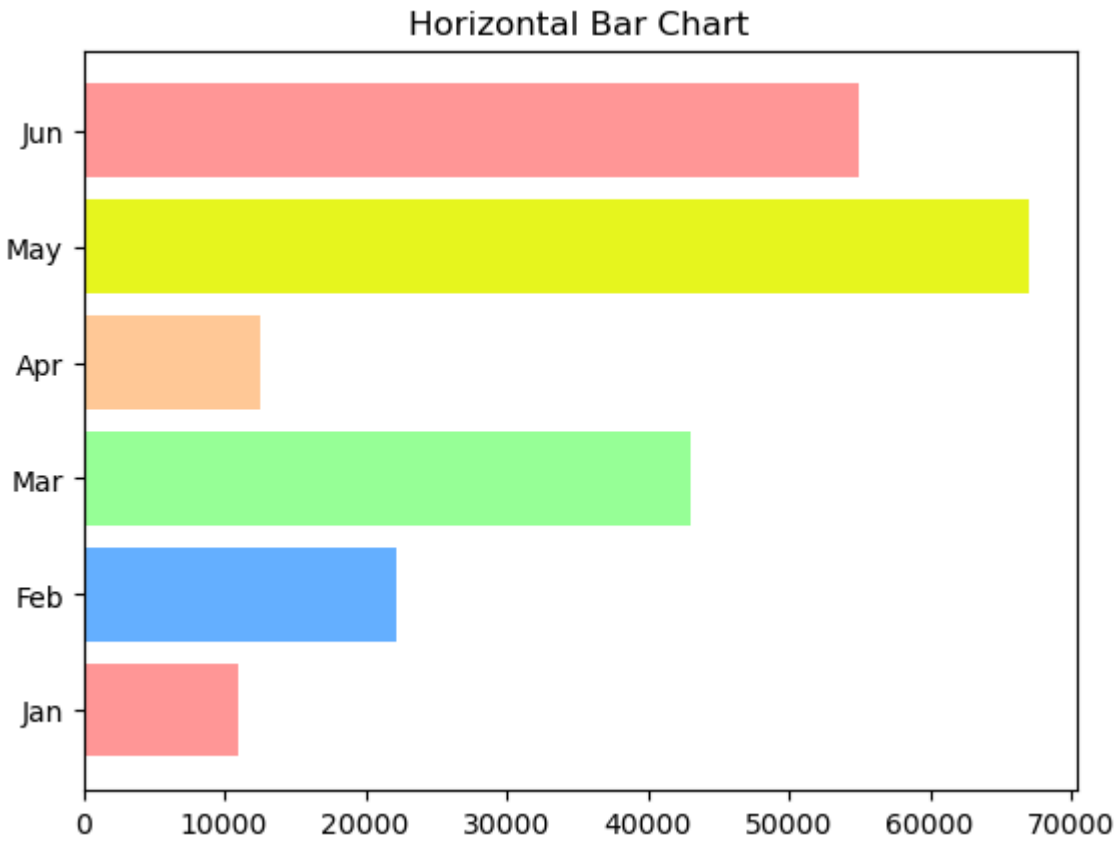


05) WAP to demonstrate the use of Bar chart.

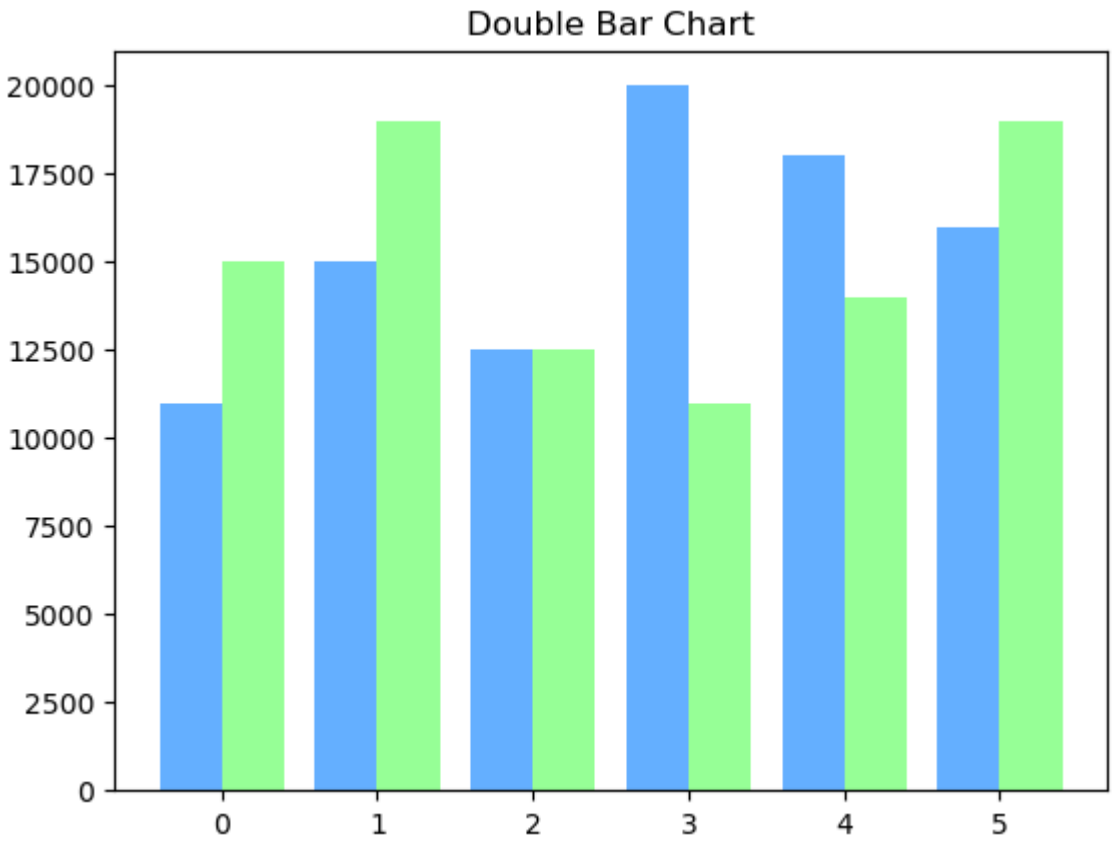
In [51]:
```
m=["Jan","Feb","Mar","Apr","May","June"]
v=[100,200,300,400,500,600]
#sufficient no aapiye to it starts repeating itself
colors = ['#ff9999','c','g','b','m','r']
plt.bar(m, v, color=colors,width=0.57)
plt.title("Bar Chart")
plt.show()
```



In [53]:
```
m = ["Jan","Feb","Mar","Apr","May","Jun"]
vis = [11000,22200,43000,12500,67000,55000]
colors = ['#ff9999', '#66b3ff', '#99ff99', '#ffcc99','#eaf520']
plt.barh(m, vis, color=colors)
plt.title("Horizontal Bar Chart")
plt.show()
```

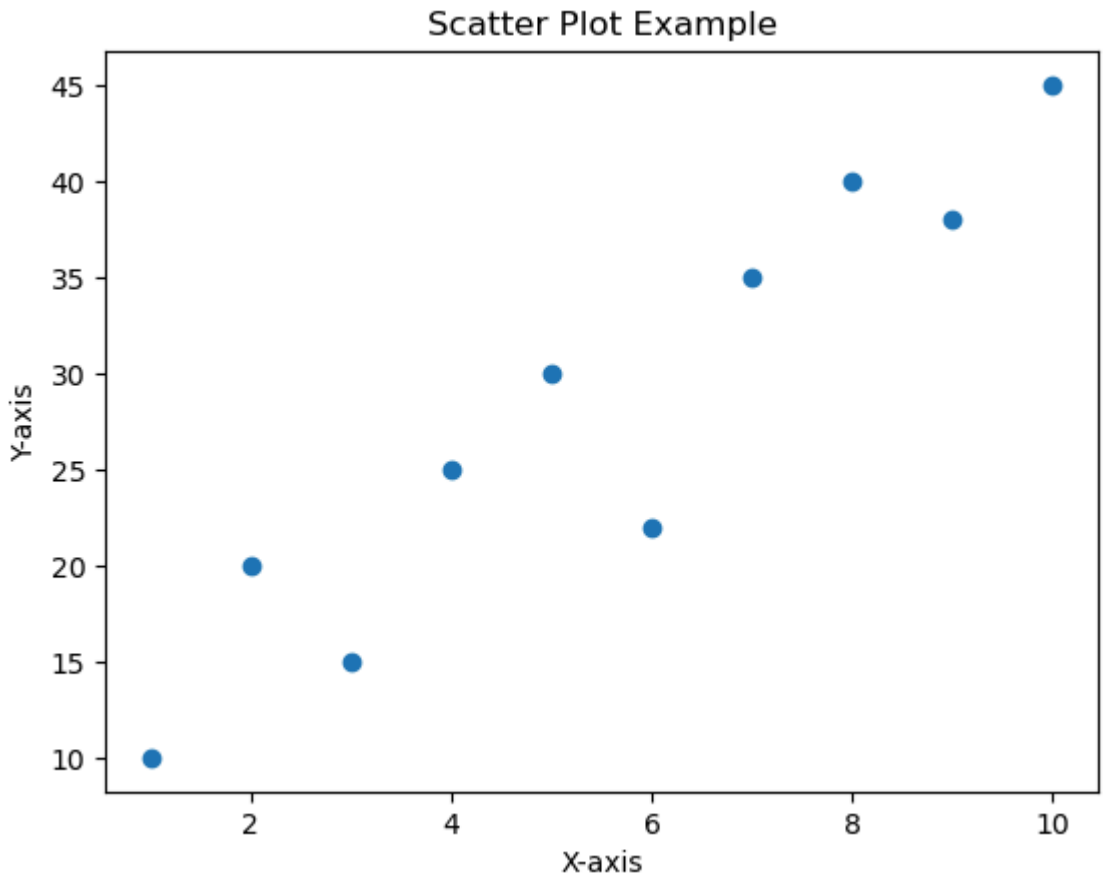


In [55]:
```
months = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun']
ggle = [11000, 15000, 12500, 20000, 18000, 16000]
yahoo = [15000,19000,12500, 11000, 14000, 19000]
x = range(len(months))
wd = 0.4
plt.bar([i-wd/2 for i in x], ggle, width=wd, color='#66b3ff')
plt.bar([i+wd/2 for i in x], yahoo, width=wd, color='#99ff99')
plt.title("Double Bar Chart")
plt.show()
```



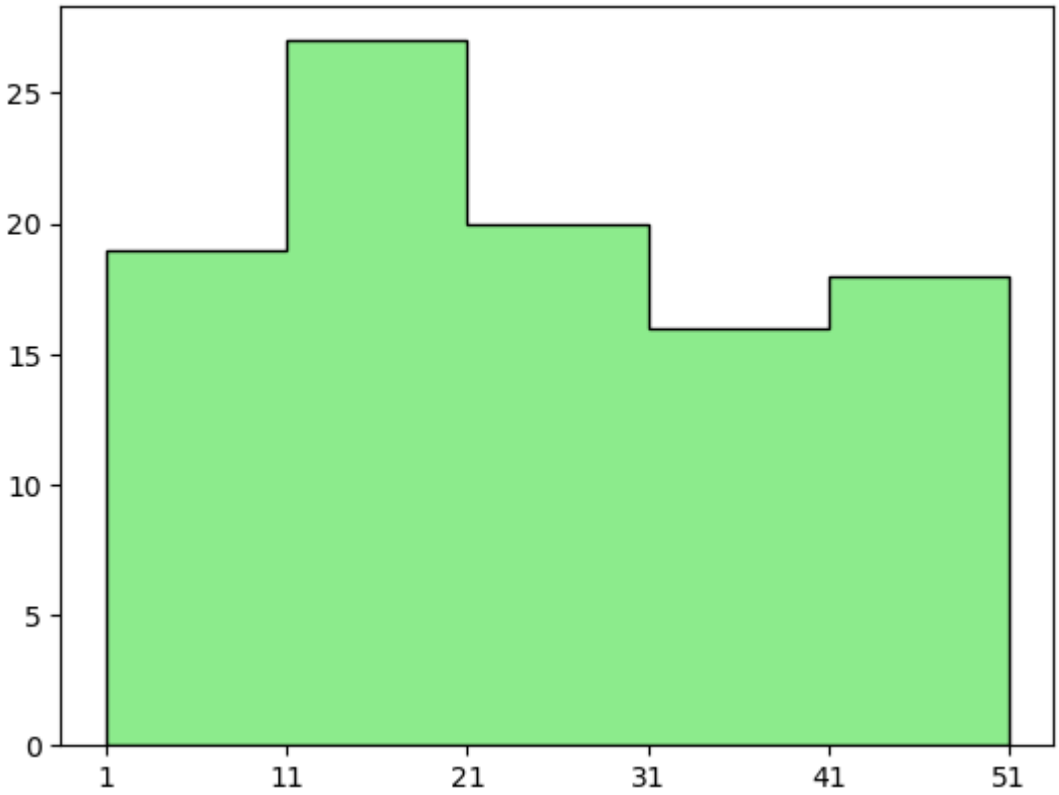
06) WAP to demonstrate the use of Scatter Plot.

In [57]:
```
x = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
y = [10, 20, 15, 25, 30, 22, 35, 40, 38, 45]
plt.scatter(x, y, marker='o')
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Scatter Plot Example")
plt.show()
```



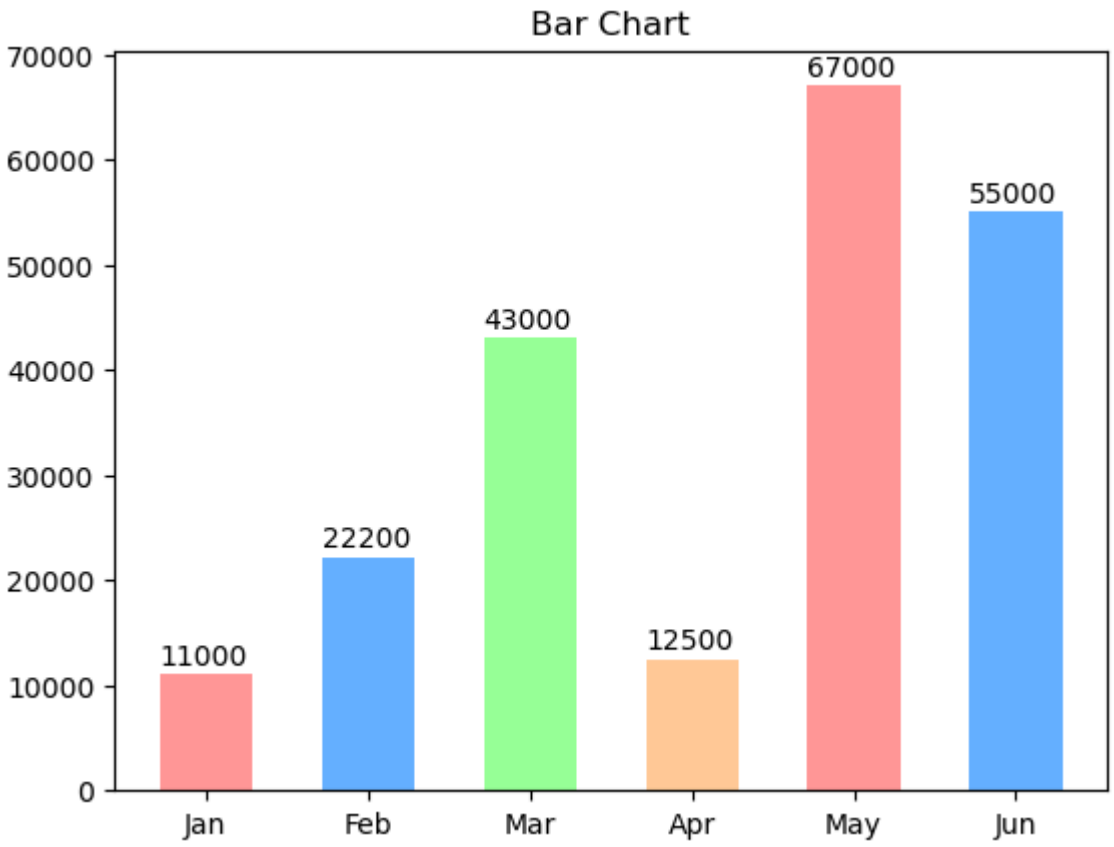
07) WAP to demonstrate the use of Histogram.

In [59]:
```
import random
random.seed(5)
age = [random.randint(1,50) for i in range(100)]
plt.hist(age, edgecolor="k", bins=[1, 11, 21, 31, 41, 51],
color="lightgreen", histtype="stepfilled")
plt.xticks([1, 11, 21, 31, 41, 51])
plt.show()
```



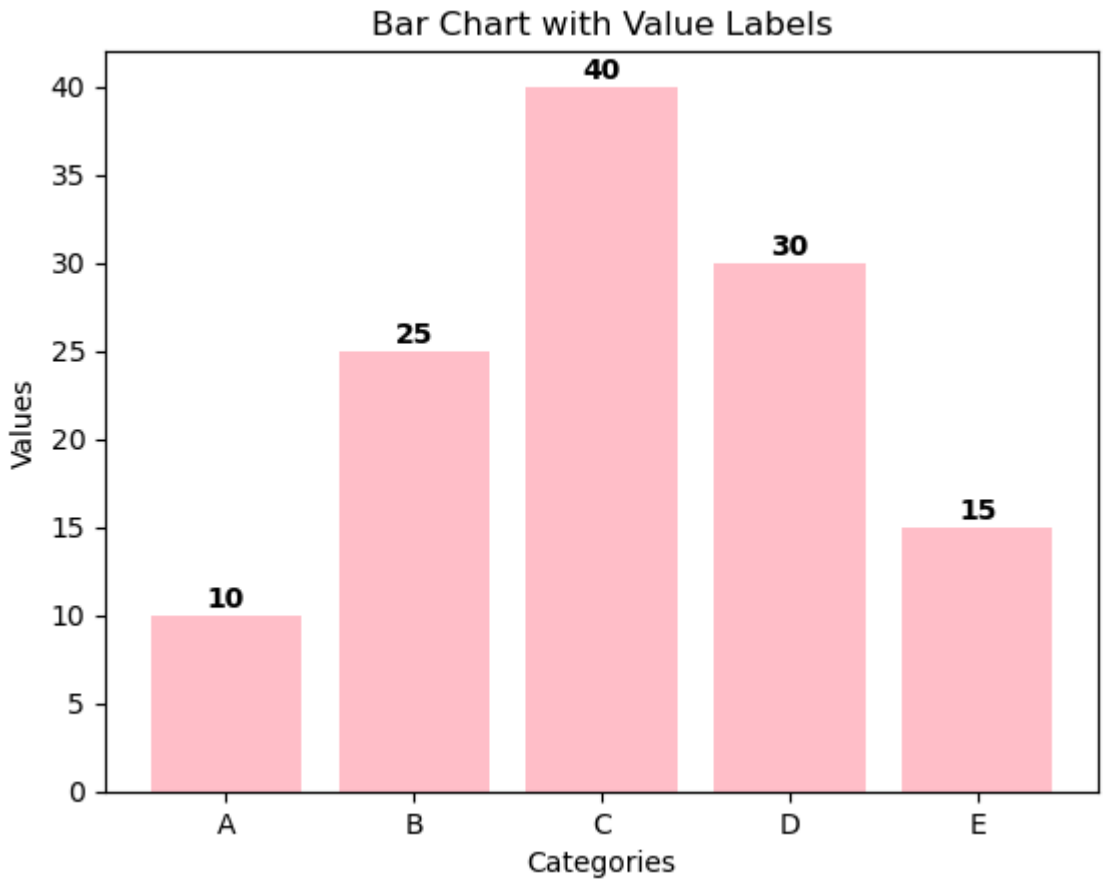
08) WAP to display the value of each bar in a bar chart using Matplotlib.

In [61]:
```
m = ["Jan","Feb","Mar","Apr","May","Jun"]
vis = [11000,22200,43000,12500,67000,55000]
#sufficient no aapiye to it starts repeating itself
colors = ['#ff9999', '#66b3ff', '#99ff99', '#ffcc99']
bars = plt.bar(m, vis, color=colors,width=0.57)
for i in bars:
 yc = i.get_height()
 plt.text(i.get_x(), yc+1000,f"{yc}")
plt.title("Bar Chart")
plt.show()
```



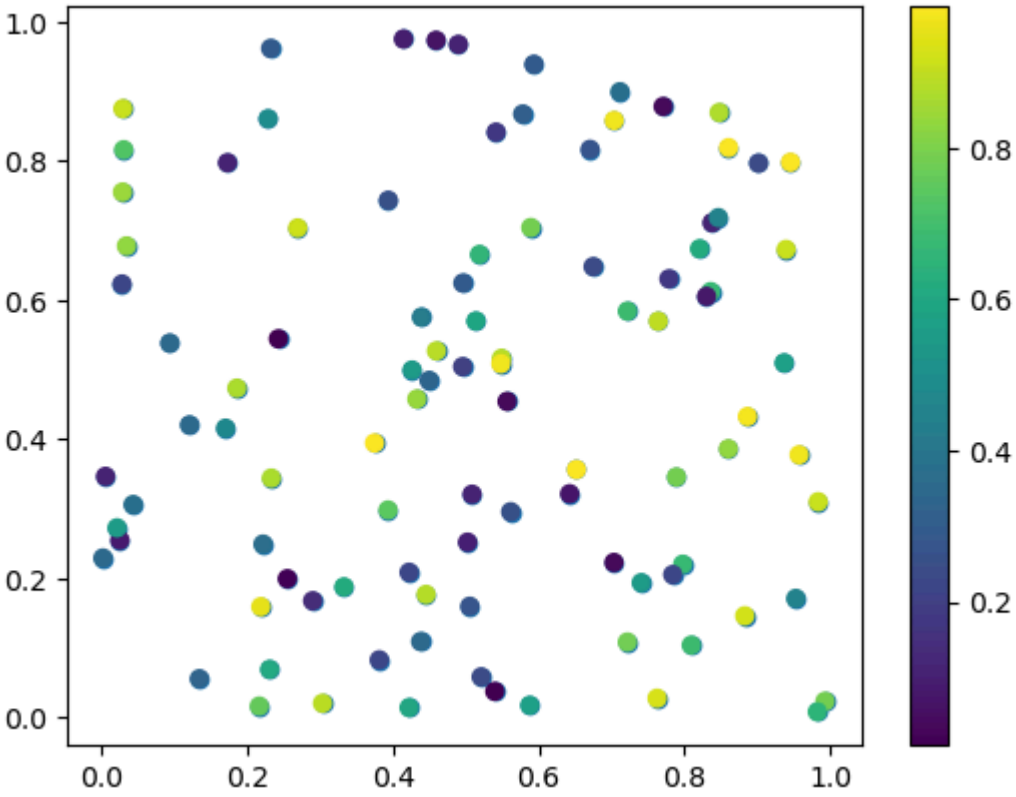
In [63]:
```
categories = ['A', 'B', 'C', 'D', 'E']
values = [10, 25, 40, 30, 15]
bars = plt.bar(categories, values, color='pink')
for i in bars :
 yc = i.get_height()
 plt.text((i.get_x() + (i.get_width() / 2)), yc+0.5, f"{yc}",
ha='center', fontweight='bold')

plt.xlabel("Categories")
plt.ylabel("Values")
plt.title("Bar Chart with Value Labels")
plt.show()
```

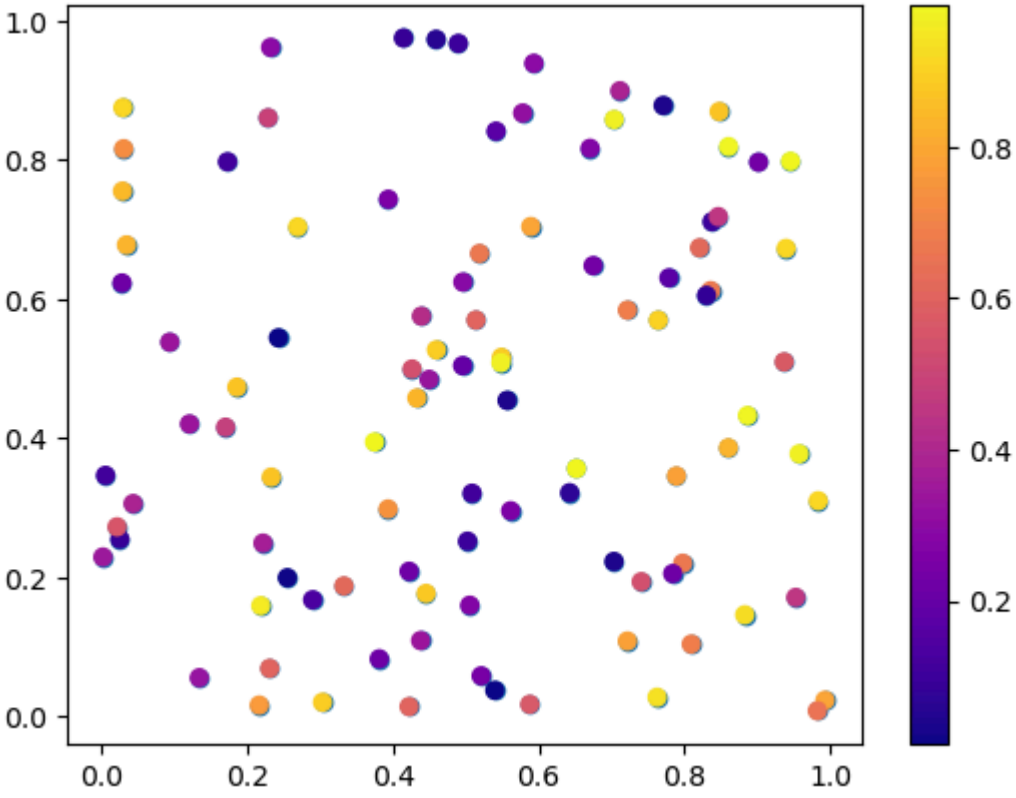


09) WAP create a Scatter Plot with several colors in Matplotlib?

In [65]:
```
random.seed(1)
x = [random.random() for i in range(100)]
y = [random.random() for i in range(100)]
z = [random.random() for i in range(100)]
plt.scatter(x, y)
plt.scatter(x, y, c=z, cmap="viridis")
plt.colorbar()
plt.show()
```

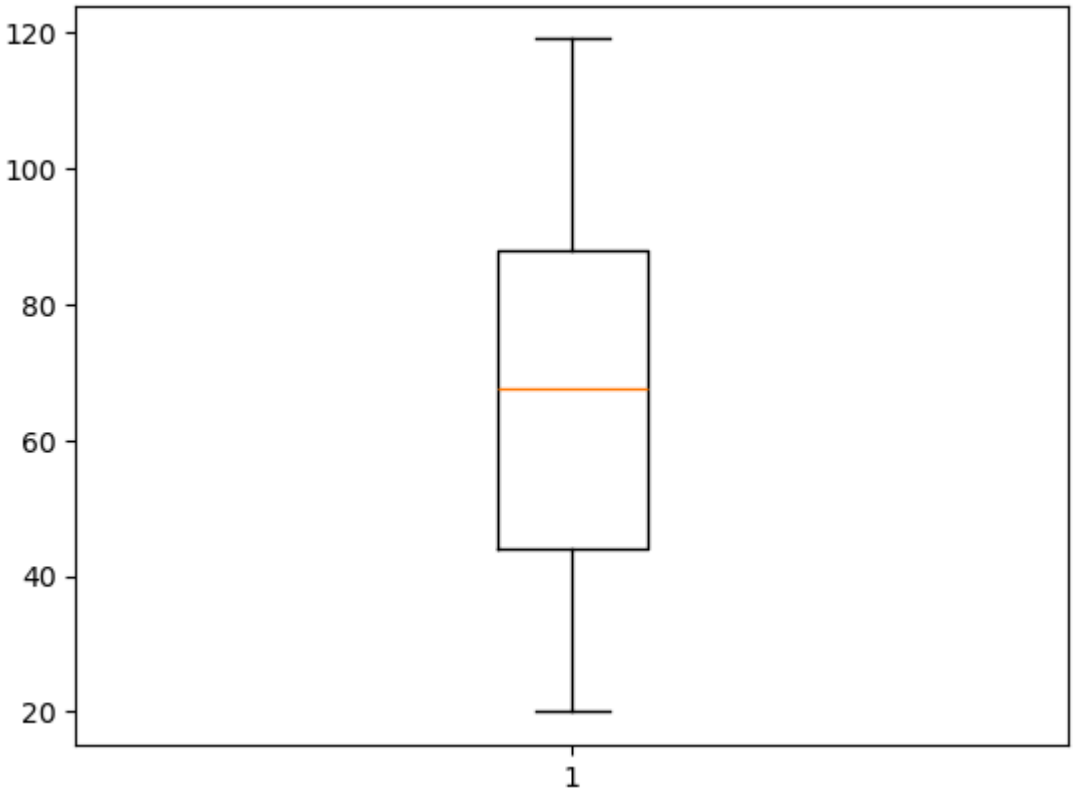


In [67]:
```
random.seed(1)
x = [random.random() for i in range(100)]
y = [random.random() for i in range(100)]
z = [random.random() for i in range(100)]
plt.scatter(x, y)
plt.scatter(x, y, c=z, cmap="plasma")
plt.colorbar()
plt.show()
```

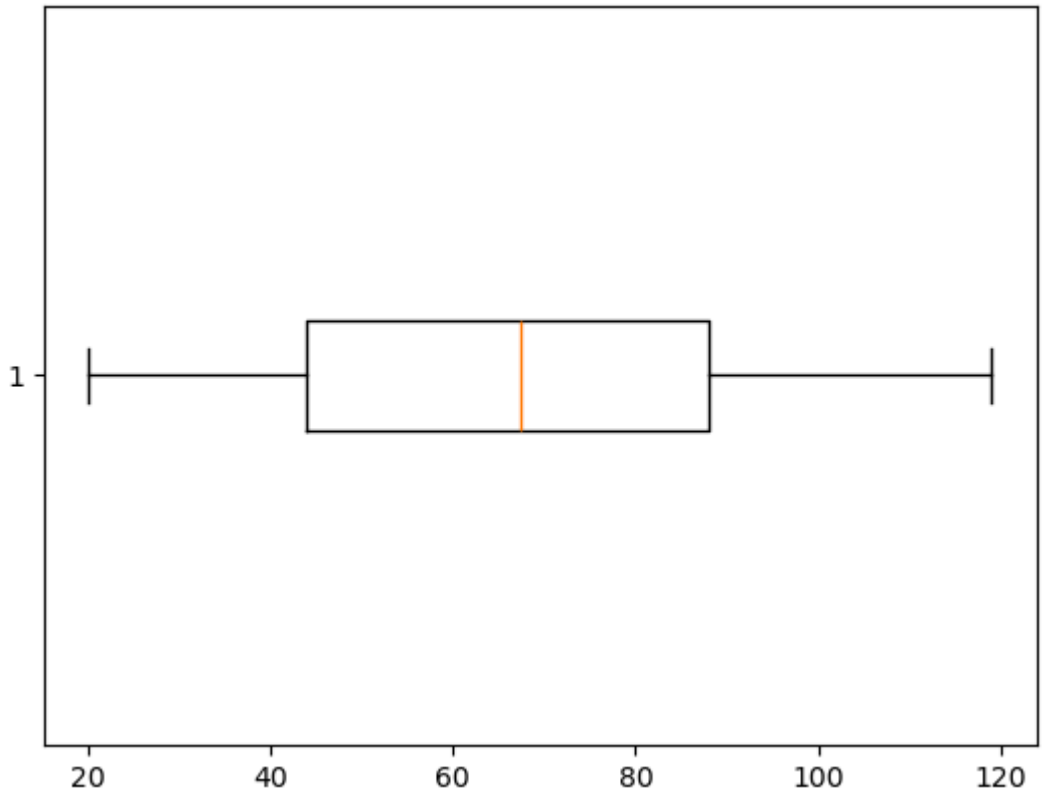


10) WAP to create a Box Plot.

In [69]:
```
random.seed(10)
time = [random.randint(20,120) for i in range(100)]
plt.boxplot(time)
plt.show()
```



In [71]:
```
random.seed(10)
time = [random.randint(20,120) for i in range(100)]
plt.boxplot(time, vert=False)
plt.show()
```



In [ ]:

Darshan UNIVERSITY
योगः कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 13

OOP

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

In [13]:
```
class student:
    def __init__(self,name,age,grade):
        self.nme=name
        self.age=age
        self.grd=grade
    def print_details(self):
        print(f"Name is: {self.nme} age is: {self.age} grade is:{self.grd}")
s1=student("Drashti",20,"A++")
s1.print_details()
```

```
Name is: Drashti age is: 20 grade is:A++
```

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

In [17]:
```
class Bank_Account:
    def __init__(self):
        pass
    def getAccountdetails(self):
        self.acno=int(input("enter the account number"))
        self.nme=input("enter the account Name")
        self.email=input("enter the EmailID")
        self.acty=input("enter the account Type")
        self.acb=int(input("enter the account balance"))
    def printDetails(self):
        print(f"accout no: {self.acno} name is: {self.nme} email is:{self.email}
ac=Bank_Account()
```

```
ac.getAccountdetails()
ac.printDetails()
```

```
accout no: 4854 name is: hb email is:vgvh Accounttype is:bjnn Accountbalance is:4
53m
```

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

In [23]:
```
class Circle:
    def __init__(self,Radius):
        self.r=Radius
    def Area(self):
        print(f"area is:{3.14*self.r*self.r}")
    def Perimeter(self):
        print(f"perimeter is:{2*3.14*self.r}")
c1=Circle(4)
c1.Area()
c1.Perimeter()
```

```
area is:50.24
perimeter is:25.12
```

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

In [42]:
```
class Employee:
    def __init__(self,name,age,salary):
        self.nme=name
        self.age=age
        self.salary=salary
    def updatedetails(self,Uname,Uage,usalary):
        #uname=None and so on if ek j update karvi hoi to ne if ma jo e none nat
         self.nme=Uname
         self.age=Uage
         self.salary=usalary
    def displaydetails(self):
        print(f"Name is: {self.nme} age is: {self.age} salary is:{self.salary}")
E1=Employee("Drashti",20,"50000000")
E1.displaydetails()
E1.updatedetails("Drashti",10,"50000000")
E1.displaydetails()
```

```
Name is: Drashti age is: 20 salary is:50000000
Name is: Drashti age is: 10 salary is:50000000
```

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

In [66]:
```
class Bank_Account:
    def __init__(self,initial_balance):
        self.ib=initial_balance
    def deposit(self,balance):
        self.ib+=balance
        print("total balance is :",self.ib)
    def withdraw(self,amount):
        if amount>self.ib:
```

```
            print("insufficient balance")
        else:
            self.ib-=amount
            print("total balance is :",self.ib)
ac=Bank_Account(5000)
ac.deposit(500)
ac.withdraw(600)
```

```
total balance is : 5500
total balance is : 4900
```

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 [80]:
```
class Manage_Inventory:
    def __init__(self):
        pass

    def addItem(self):
        self.name = input('Enter item 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('Quantity of item is: ',self.quantity)

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

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

item1 = Manage_Inventory()
item1.addItem()
print('-----------After Add-------------\n')
item1.displayItem()
print('\n-----------After Update----------\n')
item1.updateItem(quantity=3)
item1.displayItem()
print('\n----------After Delete----------\n')
item1.deleteItem()
item1.displayItem()
```

```
----------After Add--------------

Item name is:  fruit
Item price is:  12.0
Quantity of item is:  2

----------After Update----------

Item name is:  fruit
Item price is:  12.0
Quantity of item is:  3

----------After Delete----------

Item name is:  None
Item price is:  None
Quantity of item is:  None
```

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

In [84]:
```
class Student:
    def __init__(self,name,eno,age):
        self.name = name
        self.eno =eno
        self.age = age

s1=Student('Drashti Ruparelia',23010101231,19)
print('Your Name is: ',s1.name)
print('Your eno is: ',s1.eno)
print('Your age is: ',s1.age)
```

```
Your Name is:  Drashti Ruparelia
Your eno is:  23010101231
Your age is:  19
```

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 [72]:
```
class student_kit:
    p_name="Mr abc"
    def __init__(self, student_name):
        self.student_name = student_name
        self.attendance_days = 0

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

    def generate_certificate(self):
        print("\n----- Attendance Certificate -----")
```

```
        print(f"Principal: {student_kit.p_name}")
        print(f"Student Name: {self.student_name}")
        print(f"Days Present: {self.attendance_days}")
        print("Congratulations on your attendance!\n")

# Example Usage
student1 = student_kit(input("Enter Student Name: "))
days_present = int(input("Enter number of days present: "))
student1.mark_attendance(days_present)
student1.generate_certificate()
```

```
----- Attendance Certificate -----
Principal: Mr abc
Student Name: drashti
Days Present: 99
Congratulations on your attendance!
```

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

In [74]:
```
class Time:
    # constructor
    def __init__(self,h,m):
        self.h = h
        self.m = m

    #add two time object
    def addTime(self,t1,t2):
        self.h = t1.h+t2.h
        self.m = t1.m+t2.m

    #display time
    def displayTime(self):
        print(self.h,self.m)

t1 = Time(4,4)
t2 = Time(3,2)
t3 = Time(0,0)

t3.addTime(t1,t2)
t1.displayTime()
t2.displayTime()
t3.displayTime()
```

```
4 4
3 2
7 6
```

In [ ]:

Python Programming - 2301CS404

Lab - 13

Drashti Ruparelia

23010101231

Continued..

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

In [42]:
```
class Rectangle:
     def __init__(self,length, breadth):
         self.length = length
         self.breadth = breadth
     def area(self):
         return self.length*self.breadth
def calculate_area(rect):
    return rect.area()
l = float(input("Enter length: "))
b = float(input("Enter breadth: "))
rect1 = Rectangle(l,b)
print("Area of rectangle : ",calculate_area(rect1))
```

```
Area of rectangle :  6.0
```

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 [33]:
```
class Square:
    def __init__(self,length):
        self.length=length
        # self.aos=0;
    def area(self):
```

```
        aos= (self.length**2)
        self.output(aos)
    def output(self,areaofsquare):
        print("the area of square is:",areaofsquare);
s1=Square(2)
s1.area()
# s1.output()
```

```
the area of square is: 4
```

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 [58]:
```
class Rectangle :
     def __init__(self, length, breadth) :
         if length == breadth :
             print("THIS IS SQUARE")
             self.is_valid = False
         else :
             self.length = length
             self.breadth = breadth
             self.is_valid = True

     def area(self) :
         if self.is_valid==True :
             ans = self.length * self.breadth
             self.output(ans)
     def output(self,aor) :
         print("Area of Rectangle :", aor)
l = float(input("Enter the length: "))
b = float(input("Enter the breadth: "))
rect1 = Rectangle(l, b)
rect1.area()
```

```
Area of Rectangle : 6.0
```

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 [83]:
```
class Square:
    def __init__(self, side):
        # Private attribute
        self.__side = side

    # Getter method to access the private attribute
    def get_side(self):
```

```
        return self.__side

    # Setter method to modify the private attribute
    def set_side(self, new_side):
        if new_side > 0:
            self.__side = new_side
        else:
            print("Side length must be positive.")

# Example usage:
square = Square(5)
print("Initial side:", square.get_side())  # Output: 5

square.set_side(10)
print("Updated side:", square.get_side())  # Output: 10

square.set_side(-3)  # Will trigger the validation message
```

```
Initial side: 5
Updated side: 10
Side length must be positive.
```

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

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

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

In [66]:
```
class Profit :
     def getProfit(self) :
         self.profit = float(input("Enter profit : "))
class Loss :
     def getLoss(self) :
         self.loss = float(input("Enter loss : "))
class BalanceSheet(Profit, Loss) :
     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()
```

```
Net Balance : 3000.0
```

15) WAP to demonstrate all types of inheritance.

In [68]:
```
# 1. Single Inheritance
class Animal:
     def sound(self):
         print("Animals make sound")
```

```
class Dog(Animal):
     def bark(self):
         print("Dog barks")

# 2. Multiple Inheritance
class Mammal:
     def warm_blooded(self):
         print("Mammals are warm-blooded")
class Bat(Animal, Mammal):
     def fly(self):
         print("Bats can fly")

# 3. Multilevel Inheritance
class Puppy(Dog):
     def cute(self):
         print("Puppies are cute")

# 4. Hierarchical Inheritance
class Cat(Animal):
     def meow(self):
         print("Cat meows")

# 5. Hybrid Inheritance
class Fish:
     def swim(self):
         print("Fish can swim")
class Whale(Mammal, Fish):
    def big(self):
         print("Whale is the largest mammal")

print("-----")
d = Dog()
d.sound()
d.bark()
print("-----")
b = Bat()
b.sound()
b.fly()
b.warm_blooded()
print("-----")
p = Puppy()
p.sound()
p.bark()
p.cute()
print("-----")
c = Cat()
c.sound()
c.meow()
print("-----")
w = Whale()
w.swim()
w.warm_blooded()
w.big()
```

```
-----
Animals make sound
Dog barks
-----
Animals make sound
Bats can fly
Mammals are warm-blooded
-----
Animals make sound
Dog barks
Puppies are cute
-----
Animals make sound
Cat meows
-----
Fish can swim
Mammals are warm-blooded
Whale is the largest mammal
```

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 [79]:
```
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("Drashti", 20, 10000000)
emp.display()
```

```
Name : Drashti
Age : 20
Salary : 10000000
```

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 [87]:
```
from abc import ABC, abstractmethod
class Shape(ABC): # Abstract Base Class
     @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
```