Week - 1 Basic Programs

1.WAP in Python to print your name, branch, and interest on the screen. (WAP--write a python program)

In [2]:
```
name='Jayasurya'
branch='Datascience'
interest='visit temple'
#print("My name is ",name,"I am",branch ,"branch and I am interset in",intere
print(f"my name is {name},i am branch of {branch},and i am interest of {inter
```

```
my name is Jayasurya,i am branch of Datascience,and i am interest of visit temp
le
```

2.WAP in Python to print “Hi Welcome to BU username” where username is entered by the user.

In [3]:
```
name=input("Enter user name")
print("Welcome to BU",name)
```

```
Enter user namesurya
Welcome to BU surya
```

3.WAP in Python to input an integer number and display its square.

In [4]:
```
a=int(input("Enter a integer number"))
print("The square of integer number",a**2)
```

```
Enter a integer number3
The square of integer number 9
```

4.WAP in Python to input a floating number and display.

In [4]:
```
a=float(input("Enter a float values"))
print("The cube of float values is:",a**3)
```

```
Enter a float values3
The cube of float values is: 27.0
```

5.WAP in Python to input a string variable and print its type.

In [64]:
```
a=input("Enter a String")
a=type(a)
print(a)
```

```
Enter a Stringjai
<class 'str'>
```

6.WAP in Python to input length and breadth of a rectangle and print the area and perimeter

In [ ]:
```
print("Enter the length and breadth values")
length=int(input("Enter the length  values"))
breadth=int(input("Enter the breadth  values"))
print("Area of rectangle",length*breadth)
print("perimeter of rectangle",(2*length)+(2*breadth))
#simplify the formula
print("POT",(2*(length+breadth)))
```

7.WAP in Python to calculate and display area of a circle.

In [2]:
```
pi=3.14
print("Enter a r value")
radius=float(input("enter here")) #pi--3.14
print("Area of circle is",pi*radius**2)
```

```
Enter a r value
enter here3
Area of circle is 28.26
```

8.WAP in Python to find the product of two integer numbers.

In [3]:
```
a=int(input("enter a value"))
b=int(input("enter b value"))
product=a*b
print("product of a and b",product)
```

```
enter a value3
enter b value4
product of a and b 12
```

9.WAP in Python to find the sum of two floating numbers.

In [4]:
```
a=float(input("enter a value"))
b=float(input("enter b value"))
sum_of=a+b
print(sum_of)
```

```
enter a value4.5
enter b value4.6
9.1
```

10.A student gets 92, 80, 75 and 95 in English, Physics, Maths and Biology. Write a program in Python to calculate the average marks obtained by the student.

In [5]:
```
a=English,Physics,Maths,Biology=92,80,75,95
avg=sum(a)/len(a)
print("The average of student is:",avg)
```

```
The average of student is: 85.5
```

11.WAP in Python which takes a person's age to print the number of years left for retirement (A person retires at 65)

In [6]:
```
age=int(input("enter ur age"))
retair=65-age
print("The wait of retairment year",retair)
```

```
enter ur age55
The wait of retairment year 10
```

12.WAP in Python that takes Celsius temps from the user and convert it into Fahrenheitand displays the result

In [11]:
```
cel=int(input("enter the celcius values"))
print("The Fahrenheit values:",cel*(9/5)+32,"F")
```

```
enter the celcius values6
The Fahrenheit values: 42.8 F
```

13.WAP in Python to evaluate the expression: x2 + 3x+7.

In [12]:
```
x=int(input("enter x  value"))
print("expression",x*2+ 3*x+7)
```

```
enter x  value4
expression 27
```

14.WAP in Python to find the square root of a given number.

In [7]:
```
import math
x=int(input("enter x value"))
print("Square root of given number",math.sqrt(x))
```

```
enter x value81
Square root of given number 9.0
```

15. WAP in Python to find m raised to power n.

In [20]:
```
m=int(input("enter m value"))
n=int(input("enter n values"))
print("m raised to power of n",m^n)
```

```
enter m value2
enter n values3
m raised to power of n 1
```

16.WAP in Python to find the roots of a quadratic equation.

In [22]:
```
import cmath
a=float(input('Enter a:'))
b=float(input('Enter b:'))
c=float(input('Enter c:'))
d=(b**2)-(4*a*c)
s1=(-b-cmath.sqrt(d))/(2*a)
s2=(-b+cmath.sqrt(d))/(2*a)
print("The solution are {0} and{1}".format(s1,s2))
```

```
Enter a:1
Enter b:5
Enter c:6
The solution are (-3+0j) and(-2+0j)
```

17.WAP in Python which converts 8 hours and 32 minutes into seconds

In [4]:
```
print("8 hours and 32 minutes:",8*3600+32*60,'seconds')
```

```
8 hours and 32 minutes: 30720 seconds
```

18.WAP in Python that asks the user for his/her amount of money, then reports howmany items the person can buy, and how much more money he/she will need to afford an additional item (cost of each item is=500)

In [5]:
```
no_of_items=int(input("enter how many items do u want"))
cost=500
if no_of_items > 1:
    cost=no_of_items*500
else:
    cost=1*500
budget=int(input("How much money do u have"))
bal=cost-budget
print("U need  balance amount",bal)
```

```
enter how many items do u want3
How much money do u have600
U need  balance amount 900
```

19.WAP in Python to swap two integer variables using a third variable

In [27]:
```
a = 10
b = 20
print("Before swapping:", a, b)
temp=a
a=b
b=temp
print("After swapping:", a,b)
```

```
Before swapping: 10 20
After swapping: 20 10
```

20 WAP in Python to swap two integer variables without using a third variable.

In [6]:
```
a=10
b=4
print("Before swap",a,b)
a,b=b,a
print("After swap",a,b)
```

```
Before swap 10 4
After swap 4 10
```

Week - 2 Bitwise Operator and Conditional Statements

1. Take values of three integers from the user, where values of any two variables are same and the other variable is different. Write a program in python print the value which appears only once.

In [3]:
```
print("please enter any 2 values are same")
a=int(input("Enter a:"))
b=int(input("Enter b:"))
c=int(input("Enter c:"))
if(a==b)or(b==c)or(c==a):
    print("two values are same")

else:
    print("not same")
```

```
please enter any 2 values are same
Enter a:1
Enter b:2
Enter c:3
not same
```

2. Take an integer from the user. Write a program in Python to divide and multiply the value of the variable by using bitwise operator.

In [1]:
```

i=int(input("Enter integer:"))

print("The number of values is :",i>>1,i<<1)
```

```
Enter integer:7
The number of values is : 3 14
```

3. Write a program in Python to print the Kth least significant bit of a number using only bitwise operator.

In [9]:
```
def surya(num,k):
    return bool(num&(1<<(k-1)))
k=3
num=int(input("number here:"))
res=surya(num,k)
if res:
    print(1)
else:

    print(0)
```

```
number here:23
1
```

4. if-else statement: Write a program to take an integer from the user and check whether the number is odd or even.

In [15]:
```
x=int(input("enter numbers"))
if x%2==0:
    print(x,"Yes its even number")
else:
    print(x,"Its odd even number")
```

```
enter numbers2
2 Yes its even number
```

5. Write a program to take an integer from the user and check whether the number is positive or negative. ")

In [17]:
```
x=int(input("Enter integer"))
if x>=0:
    print(x,"yeah its positive number")
else:
    print(x,"Its negative number")
```

```
Enter integer200
200 yeah its positive number
```

6. Write a program to accept three numbers and check whether thefirst is between the other two numbers.

In [7]:
```
n1=int(input("enter n1 value"))
n2=int(input("enter n2 value"))
n3=int(input("enter n3 value"))

if((n1>n2)or (n1<n3))or((n1<n2) or(n1>n3)):
    print("The first number is between of second and third number")
else:
    print("The first  number is not between of second and third number")

```

```
enter n1 value78
enter n2 value68
enter n3 value90
The first number is between of second and third number
```

7. Accept a character as input and check whether the character is a digit or alphabet. (Check if it is in the range '0' to '9999999' both inclusive

In [1]:
```
ch=input("please enter ur own character")
if((ch>='a' and ch<'z') or (ch>='A' and ch<'Z')):
    print("The given character",ch,"is  an alphabet")
elif((ch>='0'and ch<='9999999')):
    print("The given character",ch,"is  an digit")
else:
    print("The given character",ch,"is  not an alphabet and digit" )
```

```
please enter ur own characterA
The given character A is  an alphabet
```

8. Accept a lowercase/ uppercase character from the user and check whether the character is a vowel or consonant.

In [2]:
```
ch=input("Please enter character")
if ((ch=='a')or (ch=='e')or(ch=='i')or(ch=='o')or(ch=='u')):
    print(ch,"its vowels")
else:
    print(ch,"Its constant")
```

```
Please enter charactera
a its vowels
```

9. Write a program, which accepts annual basic salary of an employee and calculates and displays the Income tax as per the following rules.If Basic is less than Rs. 1,50,000/-, then Tax = 0 If Basic is from Rs. 1,50,000/- to Rs. 3,00,000/-, then tax is 20% If Basic is greater than Rs. 3,00,000/-, then tax is 30%

In [1]:
```
1
2  name=input("Enter ur name")
3  sal=int(input("Enter ur monthly income"))
4  annual_income=sal*12
5  if annual_income<150000:
6      print(name,"your annual income",annual_income,"and your tax of this year
7  elif  ((annual_income>=150000) and (annual_income<300000)) :
8      print(name,"your annual income",annual_income,"and your tax of this year
9  elif annual_income>=300000 :
10     print(name,"your annual income",annual_income,"and your tax of this year
11 else:
12     print("Enter correct")
```

```
Enter ur nameJayasurya
Enter ur monthly income18400
Jayasurya your annual income 220800 and your tax of this year is 44160.0
```

10. Write a program to check whether given character is a digit or a character in lowercase or uppercase alphabet.

In [3]:
```
1  ch=input("please enter any character")
2  if(ch>='a' and ch<'z'):
3      print("The given character",ch,"is  an alphabet","it is lower case")
4  elif (ch>='A' and ch<'Z'):
5      print("The given character",ch,"is  an alphabet","It is uppercase")
6
7  elif (ch>='0'and ch<='999999'):
8       print("The given character",ch,"is  an digit ")
9  else:
10     print("The given character",ch,"is  not an alphabet and digit" )
```

```
please enter any character3
The given character 3 is  an digit
```

Another way using Ascii values Python Program to check character is Lowercase or Uppercase using ASCii values

In [3]:
```
1
2  ch = input("Please Enter Your Own Character : ")
3
4  if(ord(ch) >= 65 and ord(ch) <= 90):
5      print("The Given Character ", ch, "is an Uppercase Alphabet")
6  elif(ord(ch) >= 97 and ord(ch) <= 122):
7      print("The Given Character ", ch, "is a Lowercase Alphabet")
8  else:
9      print("The Given Character ", ch, "is Not a Lower or Uppercase Alphabet")
```

```
Please Enter Your Own Character : J
The Given Character  J is an Uppercase Alphabet
```

11. Accept any year as input through the keyboard. Write a program to

check whether the year is a leap year or not.

In [8]:
```
year=int(input("Enter any year"))
if year%4==0:
    print(year,"is Leap year")
else:
    print(year,"is not leap year")
```

```
Enter any year2020
2020 is Leap year
```

12. Accept three sides of triangle as input, and print whether the triangle is valid or not.

In [1]:
```
s1=int(input("Enter first side value of triangle"))
s2=int(input("Enter second side value of triangle"))
s3=int(input("Enter third side value of triangle"))
if s1+s2>s3:#first side + second side greater than third side
    if s2+s3>s1:#second side + thirdside greater than first side
        if s3+s1>s2:
            print("The given value is accept of shape of triangle")
        else:
            print("not valid triangle")

    else:
        print("not valid triangle")
else:
    print("not valid triangle")
```

```
Enter first side value of triangle5
Enter second side value of triangle8
Enter third side value of triangle10
The given value is accept of shape of triangle
```

13. Accept the x and y coordinate of a point and find the quadrant in which the point lies.

In [2]:

```
x=int(input("enter x value"))
y=int(input("enter y value"))
#if both x and y are positive, then the point lies in the first quadrant
if ((x>0)and (y>0)):
    print("The point lies in FIRST quadrant")
#if x is negative y is positive, then the point lies in the second quadrant
elif((x<0)and(y>0)):
    print("The point lies in SECOND quadrant")
#if  both x and y is negative, then the point lies in the third quadrant
elif((x<0)and(y<0)):
     print("The point lies in THIRD quadrant")
#if x is positive y is negative,then the point lies in the fourth quadrant
elif((x>0)and(y<0)):
    print("The point lies in FOURTH quadrant")
```

```
enter x value2
enter y value-6
The point lies in FOURTH quadrant
```

14. Accept the cost price and selling price from the keyboard. Find out if the seller has made a profit or loss and display how much profit or loss has been made.

In [3]:

```

cp=int(input("Enter the product cost price"))
sp=int(input("Enter the product selling price"))
if sp>cp:
    print("This selling is profit of seller","and the profit amount is RS.",s
else:
    print("This selling is loss of seller","and the loss amount is RS.",cp-sp
```

```
Enter the product cost price2500
Enter the product selling price2300
This selling is loss of seller and the loss amount is RS. 200
```

15. Write a program to accept marks for three subjects and find the total marks secured, average and also display the class obtained. (Class I – above 80%, class II – 60% to 80%, pass class – 40% to 59% and fail otherwise)

In [2]:

```
name=input("Enter ur name")
m1=int(input("enter Python mark"))
m2=int(input("enter statistics mark"))
m3=int(input("enter database mark"))
sum=m1+m2+m3
avg=sum//3
if avg>80:
    print("Congragulations!!",name,"you scored",sum, "ur percentage is",avg,"
elif ((avg>=60)and(avg<80)):
     print("Congragulations!!",name,"you scored",sum, "ur percentage is",avg,
elif ((avg>=40)and(avg<59)):
     print("Good!!",name,"you scored",sum, "ur percentage is",avg,"%","and u
else:
     print("Better next time!",name,"you scored",sum, "ur percentage is",avg,

```

```
Enter ur namesurya
enter Python mark99
enter statistics mark89
enter database mark90
Congragulations!! surya you scored 278 ur percentage is 92 % and u got class I
```

16. Write a program to accept quantity and rate for three items, compute the total sales amount,Also compute and print the discount as follows: (amount > Rs. 2000/- : 20% discount, amount between Rs. 1500/- to Rs.1999/- :15% discount, amount between Rs. 1000/- to Rs.1499/- 8 % discount)

In [5]:

```
name=input("enter customer name")#customer name
itm=input("enter items name")#product name
qut=int(input("enter quantity of items"))#quantity
rate=int(input("enter rate of item"))#rate of product
ta=rate*qut #total amount=rate*quantity
if ta>2000:
    print("congrats",name, "dude u purchase of",qut,itm,"you get 20% discount
elif ((ta>=1500)and (ta<=1999)):
    print("Best of luck ",name,"dude u purchase of",qut,itm," you get 15% dis
elif((ta>=1000)and (ta<=1499)):
    print("welcome ",name," u purchase of",qut,itm, "you get 8% discount so p
else:
    print("Thanks for coming",name, "u purchase of ",qut,itm,"please pay",ta)
```

```
enter customer namesurya
enter items namecasual shoe
enter quantity of items4
enter rate of item530
congrats surya dude u purchase of 4 casual shoe you get 20% discount so please
pay $. 424.0
```

17. A library charges a fine for every book returned late. Accept the number of days the member is late, compute and print the fine as

follows:(less than five days Rs. 100/- fine, for 6 to 10 days Rs. 200/- fine and above 10 days Rs. 300/- fine).

In [6]:

```
1
2  import datetime
3  rno=int(input("Enter ur roll no"))
4  bname=input("Enter bought of book name")
5  days=int(input("enter the no of delay days of return ur book"))
6  if days<5:
7      print(rno,"you return",bname,days,"delay so u pay fine $100")
8  elif ((days>=6) and (days<=10)):
9       print(rno,"you return",bname,days,"delay so u pay fine $200")
10 elif days>10:
11      print(rno,"you return",bname,days,"delay so u pay fine $300")
12 else:
13     print(rno,"u delay lot of days so please meet librarian sir!!")
14
```

```
Enter ur roll no165114192
Enter bought of book nameBanking GK
enter the no of delay days of return ur book7
165114192 you return Banking GK 7 delay so u pay fine $200
```

Evaluation of expressions

Solve the expression given below:

1. Evaluate X1=(10+35+23+10+7+5)/((1-(1/2)-(1/20)))

```
X1=(10+35+23+10+7+5)/((1-(1/2)-(1/20)))

X1=(10+35+23+10+7+5)/(1-0.5)-(0.05)

X1=(10+35+23+10+7+5)/(0.5-0.05)

X1=90/0.45
X1=200.0
```

In [1]:
```
#1.program
X1=(10+35+23+10+7+5)/((1-(1/2)-(1/20)))
print(X1)
```

```
200.0
```

2. X2=(((10*9)+8-((7//6)%(5*4)))&3)|(2<<1)

```
 2. X2=(((10*9)+8-((7//6)%(5**4)))&3)|(2<<1)

X2=(((90)+8-((1)%(625)))&3)|(2<<1)
X2=(97%625)&3|(2<<1)
X2=(97&3)|(4)
x2=5
```

In [2]:
```
#2.program

X2=(((10*9)+8-((7//6)%(5**4)))&3)|(2<<1)

print(X2)
```

```
5
```

3. P=20*1+100*2+6*4+3*8 X3=(P-(118*2))/4

```
3. P=20*1+100*2+6*4+3*8 X3=(P-(118*2))/4

P=20+200+24+24
P=268
X3=(268-(236))/4
```

```
X3=32/4
X3=8.0
```

In [4]:

```
#3.program
P=20*1+100*2+6*4+3*8
X3=(P-(118*2))/4
print(X3)


```

```
8.0
```

Week - 3 Loops and Functions

1.Write a program that accepts numbers continuously as long as the number is positive and prints the sum of the numbers read.

In [1]:
```
sum=0
while True:

    a=int(input("Enter some values here"))
    sum+=a;
    print('sum:',sum)
    if a>=0:
        break
```

```
Enter some values here-2
sum: -2
Enter some values here-9
sum: -11
Enter some values here-6
sum: -17
Enter some values here-5
sum: -22
Enter some values here0
sum: -22
```

2. Write a program to accept n and display its multiplication table. Value of n must be provided by the user.(Example: n*1, n*2,....,n*10)

In [2]:
```
n=int(input("enter a value"))
for i in range(1,11):
    print(i,'*',n,'=',n*i)
```

```
enter a value3
1 * 3 = 3
2 * 3 = 6
3 * 3 = 9
4 * 3 = 12
5 * 3 = 15
6 * 3 = 18
7 * 3 = 21
8 * 3 = 24
9 * 3 = 27
10 * 3 = 30
```

3. Write a program to accept characters till the user enters null and count number of times an alphabet (b, where b can be any alphabet) is entered.

In [1]:
```
count=0
while True:
    b=input("enter any alphabet")
    if b=='null':
        print("The count of strings",count,"letters")
        break


    print(b)
    count+=1
```

```
enter any alphabetj
j
enter any alphabetk
k
enter any alphabetl
l
enter any alphabetm
m
enter any alphabetnull
The count of strings 4 letters
```

4.Write a program to take the values of two integers m and n from the user and display and calculate the sum of even number between m and n (including both m and n). Please note that value of m must be less than value of n.

In [5]:
```
m=int(input("enter the start"))
n=int(input("enter the limit"))
sum=0
while m<n:
    if m%2==0:
        sum=sum+m
    m=m+1
    print("m num is",m,"n num is",n )
    print("sum of even number",sum
          )
```

```
enter the start1
enter the limit6
m num is 2 n num is 6
sum of even number 0
m num is 3 n num is 6
sum of even number 2
m num is 4 n num is 6
sum of even number 2
m num is 5 n num is 6
sum of even number 6
m num is 6 n num is 6
sum of even number 6
```

5. Write a program to take the values of two integers m and n from the user andcalculate mn. Do not use exponent operator (**) or pow().

m=base,n=exp

In [6]:
```
def power(base,exp):
    if (exp==1):
        return(base)
    if (exp!=1):

        return(base*power(base,exp-1
                          ))
base=int(input("enter m value"))
exp=int(input("enter n value"))



print("result:",power(base,exp))
```

```
enter m value2
enter n value3
result: 8
```

6.Write a program to take an integer from the user and check if it is prime or not.

without using function

In [16]:
```
num=int(input("enter number"))
if num>1:
    for i in range(2,num//2):
        if (num%i)==0:
            print(num,"is not a prime number")

            break
        else:
            print(num,"is a prime number")

else:
    print(num,"is not a prime number")
```

```
enter number11
11 is a prime number
11 is a prime number
11 is a prime number
```

Using function

In [3]:

```
def prime(n):
    if(n==1):
         return False
    elif(n==2):
        return True
    else:
        for x in range(2,n):
            if(x%n==0):
                return False
            return True



prime(int(input(" ")))

```

```
 3
```

Out[3]: True

Another method

In [48]:

```
a=int(input("enter number:"))
k=0
for i in range(2,a//2):
    if(a%i==0):
        k=k+1
if(k<=0):
    print(a,"is prime number")
else:
    print(a,"is not prime number")
```

```
enter number:14
14 is not prime number
```

7.Write a program to take an integer from the user and count the numberof digits in the number. Do not use len() function.

In [56]:

```
a=int(input("enter a number"))
count=0
while(a>0):
    count=count+1
    a=a//10
print("The number of digits in number are:",count)

```

```
enter a number5678899
The number of digits in number are: 7
```

8.Write a program to take an integer from the user and reverse the number. (Example: I/P 123, O/P: 321)

In [57]:
```
a=int(input("enter number:"))
reverse=0
while(a>0):
    rem=a%10
    reverse=(reverse*10)+rem
    a=a//10
    print("\n Reverse of entered number is=%d" %reverse)

```

```
enter number:123

 Reverse of entered number is=3

 Reverse of entered number is=32

 Reverse of entered number is=321
```

9.Write a program to display the first n Fibonacci numbers. Value of n must be taken from the user.

In [58]:
```

#way1

n=int(input("enter n values:"))
#initialize of first and second values of series
i=0
f1=0
f2=1

#find & displaying fibonacci series

while(i<n):
    if(i<=1):
        Next=i
    else:
        Next=f1+f2
        f1=f2
        f2=Next
    print(Next)
    i=i+1
```

```
enter n values:4
0
1
1
2
```

Using function

In [60]:
```
#way2
def fibonacci(number):
    if (number==0):
        return 0
    elif(number==1):
        return 1
    else:
        return(fibonacci(number-2)+fibonacci(number-1))
number=int(input("Enter the range Number"))
for num in range(0,number):
    print(fibonacci(num))

```

```
Enter the range Number7
0
1
1
2
3
5
8
```

10. Write a program which takes an integer from the user and checks if the number is a palindrome or not. number palindrome

In [61]:
```

i=int(input("enter a values"))
rev=0
temp=number
while(temp>0):
    rem=temp%10
    rev=(rev*10)+rem
    temp=temp//10
if (i==reverse):
    print(i,"is a palindrome")
else:
    print(i,"is not a palindrome")

```

```
enter a values121
121 is not a palindrome
```

string palindrome

In [65]:

```
string=input("please enter ur string")
if(string==string[::-1]):
    print("This is a palindrome")
else:
    print("This is not palindrome")
```

```
please enter ur stringamma
This is not palindrome
```

using function

In [71]:

```
def reverse(str1):
    if(len(str1)==0):
        return str1
    else:
        return reverse(str1[1:])+str1[0]
string=input("please enter ur string")
str1=reverse(string)
print("The reverse order",str1)
if (string==str1):

        print("This is a palindrome")
else:
        print("This is not a palindrome")
```

```
please enter ur stringamma
The reverse order amma
This is a palindrome
```

11. Write a program which takes an integer from the user and checks if the number is an Armstrong number or not and take to another integers (m and n) from the user and displays all Armstrong numbers ranges from m to n. Please note that value m must be less than value of n.

In [1]:
```
#way1
num=int(input("enter the number to check amstrong"))
#initialize
sum=0
times=0
#calculating number of individual digit

Temp=num
while Temp>0:
    times=times+1
    Temp=Temp//10

# fining amstrong number
Temp=num
while Temp>0:
    Rem=Temp%10
    sum=sum+(Rem**times)
    Temp//=10

if num==sum:
    print(num,"is a amstrong number")
else:
    print(num,"is not a amstrong number")

```

```
enter the number to check amstrong4
4 is a amstrong number
```

In [3]:
```
#way2
num=int(input("enter number:"))
order=len(str(num))
sum=0
temp=num
while temp>0:
    digit=temp%10
    sum+=digit**order
    temp//=10
if num==sum:
    print(num,"is a amstrong number")
else:
    print(num,"is not amstrong number")

```

```
enter number:153
153 is a amstrong number
```

12. Consider that there are 4 subjects for a student to study: Physics, Maths, CP and

Graphics.Input the marks of 5 students in each of the 4 subjects.Write a program in Python to find total marks for each student.

In [5]:
```
for i in range(5):

    n=input("enter ur name:")
    print("please enter ur score")
    m1=int(input('Physics'))
    m2=int(input('Maths'))
    m3=int(input('CP'))
    m4=int(input('Graphics'))
    sum=m1+m2+m3+m4
    print(f"The canditate  {n},that person got total marks {sum}")
```

```
enter ur name:jai
please enter ur score
Physics89
Maths90
CP78
Graphics90
The canditate  jai,that person got total marks 347
enter ur name:surya
please enter ur score
Physics90
Maths90
CP89
Graphics90
The canditate  surya,that person got total marks 359
enter ur name:Hari
please enter ur score
Physics90
Maths98
CP97
Graphics67
The canditate  Hari,that person got total marks 352
enter ur name:Kana
please enter ur score
Physics89
Maths90
CP89
Graphics78
The canditate  Kana,that person got total marks 346
enter ur name:Shiva
please enter ur score
Physics89
Maths90
CP78
Graphics90
The canditate  Shiva,that person got total marks 347
```

13.Suppose there is an election scheduled in which 20 people are to participate. Assume that each person is assigned the id: voter_1, voter_2….voter_5 in this sequence. A person is eligible for voting if he/she has age 18 years or more. Write a program in Python to print the voter-ids for people who are eligible for voting

In [10]:

```
for i in range(5):

    vid=int(input("enter ur voter id"))
    age=int(input("enter ur age"))
    if age>=18:
        print("TN",vid,"you are eligible for vote")
    else:
        print("not eligible")
```

```
enter ur voter id1
enter ur age18
TN 1 you are eligible for vote
enter ur voter id2
enter ur age12
not eligible
enter ur voter id3
enter ur age23
TN 3 you are eligible for vote
enter ur voter id4
enter ur age45
TN 4 you are eligible for vote
enter ur voter id5
enter ur age56
TN 5 you are eligible for vote
```

14.The prime factors of 13195 are 5, 7, 13 and 29. What is the largest prime factor of the number 60085?

In [14]:

```
 #way1

n=int(input("enter num"))
for i in range(2,n+1):
    if(n%i==0):
        isprime=1
        for j in range(2,(i//2+1)):
            if(i%j==0):
                isprime=0
                break
        if (isprime==1):

            print("prime factor",i,n)
```

```
enter num60085
prime factor 5 60085
prime factor 61 60085
prime factor 197 60085
```

using function

In [15]:
```
def larprime(num):
    i=2
    while i*i <num:
        while num %i==0:
            num=num/i
        i=i+1
    return num
print(larprime(60085))
```

```
197.0
```

15.Write a program to print bilateral right angle triangle of given length.

In [42]:
```
import math
width=float(input('please enter of width of right angle'))
height=float(input('please enter of height of right angle'))
area=0.5*width*height
c=math.sqrt((width*width)+(height*height))
per=width+height+c
print("area", area)
print("Third side",c)
print("perimeter",per)
```

```
please enter of width of right angle3.4
please enter of height of right angle7.8
area 13.26
Third side 8.508818954473059
perimeter 19.70881895447306
```

In [44]:
```
from math import sqrt
print("Input")
a=float(input("a:"))
b=float(input("b:"))
c=sqrt(a**2 + b**2)
print("The lenght of",c)
```

```
Input
a:3.2
b:4.5
The lenght of 5.52177507691141
```

Week - 4 FILE PROCESSING

1.Write a python program read an entire text file.

In [4]:
```
with open("d:\\surya.txt")as s:
    content = s.read()
    print(content)
```

```
Machine learning is the scientific study of algorithms and statistical models t
hat computer systems
use to perform a specific task without using explicit instructions,
relying on patterns and inference instead. It is seen as a subset of artificial
intelligence.
```

2.Write a python program to read first n lines of a file.

In [8]:
```
with open("d:\\read.txt")as s:
    for i in range(4):
        print(s.readline())
```

```
What is Python? Executive Summary

Python is an interpreted, object-oriented, high-level programming language with
dynamic semantics.

Its high-level built in data structures, combined with dynamic typing and dynam
ic binding, make

it very attractive for Rapid Application Development, as well as for use as a s
cripting or glue
```

3.write a python program to append text to a file and display the text.

In [3]:
```
with open("d:\\surya.txt",'a')as s:
    s.write("helo suryajai")
    s.close()
s=open("d:\\surya.txt",'r')
print(s.read())

```

```
Machine learning is the scientific study of algorithms and statistical models t
hat computer systems
use to perform a specific task without using explicit instructions,
relying on patterns and inference instead. It is seen as a subset of artificial
intelligence.helo suryajaihelo suryajaihelo suryajaihelo suryajaihelo suryajaih
elo suryajai
```

4.Write a python program to read last n lines of a file.

In [6]:
```
listed=[]
with open("d:\\read.txt",'r')as s:
    for line in s:
        listed.append(line)
print(listed[-2:])

```

```
['\n', 'it very attractive for Rapid Application Development, as well as for us
e as a scripting or glue ']
```

5.Write a python program to read line by line store into list.

In [17]:
```
lists=[]
with open("d:\\read.txt")as a:
    for line in a:
        lists.append(line)
print(lists)

```

```
['What is Python? Executive Summary\n', '\n', 'Python is an interpreted, object
-oriented, high-level programming language with dynamic semantics. \n', '\n',
'Its high-level built in data structures, combined with dynamic typing and dyna
mic binding, make \n', '\n', 'it very attractive for Rapid Application Developm
ent, as well as for use as a scripting or glue ']
```

6.Write a python program to read line by line store into variable.

In [8]:
```
with open("d:\\text.txt")as s:
    a = s.read()
    print(a)
```

```
hariminion
surya shinchan
renu tom and jerry
shankar harrypotter
swetha uma
```

7.Write a python program to read line by line store into an array.

In [9]:
```
a=[]
with open("d:\\vi.txt")as s:
    for line in s:
        a.append(line)
    print(a)
```

```
['chennai\n', 'valayapatty\n', 'Trichy\n', 'Namakkal\n', 'Salem\n', 'Karur']
```

8.Write a python program to find the longest word.

In [11]:
```
def longest_word(filename):
    with open("d:\\vi.txt", 'r') as infile:
              words = infile.read().split()
    max_len = len(max(words))
    return [word for word in words if len(word) == max_len]

print(longest_word('d:\\vi.txt'))

```

```
['valayapatty']
```

9. Write a python program to count the number of lines text file.

In [3]:
```
count = 0
with open("gu.txt", "r") as f:
    for line in f:
        count += 1
        print(line, end="")
print('This file contains ', count, ' lines')
```

```
jaisurya
hariharan
shankar
charles
jai
guru
bhavanThis file contains  7  lines
```

10. Write a python program to count the frequency of words in file.

In [19]:
```
file=open("gu.txt","r+")

wordcount={}

for word in file.read().split():
    if word not in wordcount:
        wordcount[word] = 1
    else:
        wordcount[word] += 1

for k,v in wordcount.items():
    print (k, v)
```

```
jaisurya 4
hariharan 1
shankar 1
charles 1
jai 1
guru 1
bhavan 1
```

Another way of count the frequency word in file using counter

In [24]:
```
1
2 from collections import Counter
3
4 with open("gu.txt")as f:
5     wordc=Counter(f.read().split())
6 print(wordc)
```

```
Counter({'jaisurya': 4, 'hariharan': 1, 'shankar': 1, 'charles': 1, 'jai': 1,
'guru': 1, 'bhavan': 1})
```

11. Write a python program to get the file size in plain file.

In [26]:
```
1 import os
2 size = os.path.getsize('gu.txt')
3 print('Size of file is', size, 'bytes')
```

```
Size of file is 86 bytes
```

12. Write a python program to write a list to a file.

In [47]:
```
1 list=['jaisurya','hari','shankar','swetha','renu','rama']
2 for r in list:
3     with open("gu.txt","w")as f:
4         d=f.write(r)
5     print(r)
6
```

```
jaisurya
hari
shankar
swetha
renu
rama
```

13. Write a python program to copy the file to another file.

In [57]:
```
1 from shutil import copyfile
2 copyfile("gu.txt","gur.txt")
```

Out[57]:
```
'gur.txt'
```

In [58]:
```
with open("gu.txt","r") as f,open("gur.txt","w") as f1:
     print(f1.write(f.read()))
with open("gur.txt","r") as f3:
     print(f3.read())

```

```
38
jaisurya
hari
shankar
swetha
renu
rama
```

14.Write a python program to combine each line from first file with the corresponding line in second line.

In [75]:
```
with open("gu.txt")as f,open("bha.txt")as f1:
    for line1,line2 in zip(f,f1):
        print(line1+line2)
```

```
jaisurya
bishop

hari
ramakrishna

shankar
jai
```

15.Write a python program to read a random line from a file

In [20]:
```
import random
with open("d:\\gu.txt", "r") as f:
    g=f.read().splitlines()
    print(random.choice(g))

```

```
use to perform a specific task without using explicit instructions,
```

16.Write a python program to asses if a file is closed or not.

In [145]:
```
with open("gur.txt", "r") as f:
    print(f.closed)
    f.close()
    print(f.closed)
```

```
False
True
```

17.Write a python program to remove newline character from a file and write into another file.

In [24]:
```
def remove_newlines(fname):
    flist = open(fname).readlines()
    return [s.rstrip('\n') for s in flist]

print(remove_newlines("d:\\vi.txt"))

```

```
['chennai', 'valayapatty', 'Trichy', 'Namakkal', 'Salem', 'Karur']
```

Week - 5 String Functions

1. Develop a function front_back that takes in a string and return a new string where the first and last characters have been changed

In [1]:
```
def front_back(string):
    return string[-1:] + string[1:-1] + string[:1]
string=input("Enter the string:")
print(front_back(string))
```

```
Enter the string:welcome
eelcomw
```

2. Develop a function count(string, search) that returns the number of times search character appears in string.

In [4]:
```
def countr(word, search):
        return count
word = input("Enter the words to search : ")
search = input("Enter the character to search : ")
count = 0

for char in word:
    if char == search:
        count += 1

print(countr(word, search))
```

```
Enter the words to search : welcome to learn python
Enter the character to search : z
0
```

3. Write a function count_vowels(string) that counts the #'s of vowels in a string (A,E,I,O,U). Use your character counter function to implement your new function.

In [5]:
```
def count_vowels(string):
    count=0
    vowels="aeiou"
    for i in string:
        if i in vowels:
            count +=1
    print(count)
```

In [6]:
```
string = input("Enter the string to count the vowels:")
print(count_vowels(string))
```

```
Enter the string to count the vowels:welcome to learn python
7
None
```

4. Develop a function find(string, char) that returns the position of char if exists, otherwise -1

In [7]:
```
def find(string, char):
    return char
data = input("Enter the strings:")
search = input("Enter the character to find:")
s = data.find(search)
print(s)
```

```
Enter the strings:welcome to learn python
Enter the character to find:l
2
```

5. [Character Analysis] Write a program that counts the # of spaces, digits, vowels and consonants in a string that the user inputs.

In [8]:
```
def character_count(string):
    vowels = 0
    consonants = 0
    digits = 0
    spaces = 0
    for i in range(0, len(string)):
        ch = string[i]
        if ((ch >= 'a' and ch <= 'z') or (ch >= 'A' and ch <= 'z')):
            ch = ch.lower()
            if (ch == 'a' or ch == 'e' or ch == 'i' or ch == 'o' or ch == 'u'
                vowels +=1
            else:
                consonants +=1
        elif ( ch >= '0' or ch <= '9' ):
            digits += 1
        else:
            spaces += 1
    print("vowels = ",vowels, "consonants = ",consonants, "digits =",digits,
```

In [9]:
```
string = "Sally Sells 1000 sea shells"
character_count(string)
```

```
vowels =  5 consonants =  14 digits = 8 spaces =  0
```

6. Develop a function remove*punctuation(string) that returns the string after removing the following punctuations. Punctuation List = "!\"#$%&'()*+,-./:;<=>?@[\]^`{|}~"*

In [11]:

```
lists = "!\"#$%&'()*+,-./:;<=>?@[\\]^_`{|}~"
string =  input("enter the string: ")
no_punct = " "
for char in string:
    if char not in lists:
        no_punct = no_punct + char

print(no_punct)
```

```
enter the string: "hello--he --=went to ther park
 hellohe went to ther park
```

7. [Pig Latin Translator] Write a program that asks the user for a word. Translate their word into Pig Latin. A Pig Latin word can be generated using the following rules: a. Remove the first letter of the word, b. Place the first letter of the word at the end of the word, c. Add the string ”ay” to the end of the word

In [10]:

```
def front_back(string):
    return string[1:] + string[0] + 'ay'
string=input("Enter the string:")
print(front_back(string))
```

```
Enter the string:bishop
ishopbay
```

Week - 6 Tuples

Tuple assignment or packing

In [1]:
```
student = (1, 'Peter', 'Trichy')
```

In [2]:
```
print(student)
```

```
(1, 'Peter', 'Trichy')
```

In [3]:
```
# indexed by integer value
print(student[1])
```

```
Peter
```

In [5]:
```
# Immutable, you can modify
#student[1] = 'Rex'
```

In [6]:
```
pgstudent = () #empty tuple
```

In [49]:
```
temperatures = (40, 39, 40, 38, 35, 36, 41)
#slice it
wed_fri = temperatures[3:6]
print(wed_fri)
```

```
(38, 35, 36)
```

Tuple Unpacking

In [7]:
```
(rollno, name, city) = student #equalant to multiple assignment statements
```

In [8]:
```
print('Roll No: ', rollno, ' Name: ', name, ' City: ', city)
```

```
Roll No:  1  Name:  Peter  City:  Trichy
```

In [11]:
```
# swapping values of two variables
a = 10
b = 20
(a, b) = (b, a)
# a, b = b, a
print(a, b)
```

```
20 10
```

Tuples can return multiple values

In [14]:
```
import math

def circle_stats(r):
    """ Return (circumference, area) of a circle of radius r """
    circumference = 2 * math.pi * r
    area = math.pi * r * r
    return (circumference, area)
```

In [20]:
```
# call it
radius = 10.0
(c, a) = circle_stats(radius)
print('Circumference: %.3f Area: %.3f' %(c,a))
```

```
Circumference: 62.832 Area: 314.159
```

Tuple items can be tuples too

In [21]:
```
student = (1, 'Tom', (85, 90, 95))
```

In [22]:
```
print(student)
```

```
(1, 'Tom', (85, 90, 95))
```

Function to find average marks

In [29]:
```
def stud_avg(student):

    #unpack it
    roll, name, marks = student
    m1, m2, m3 = marks

    avg = (m1+ m2 + m3)/3
    print("Roll No: ", roll, ' Name:', name, ' Average: ', avg)

```

In [30]:
```
# call it
stud = (1, 'Sam', (85, 90, 95))
stud_avg(stud)
```

```
Roll No:  1  Name: Sam  Average:  90.0
```

Tuple item can be a list

In [31]:
```
student = (1, 'Rex', [85, 90, 95])
```

In [32]:
```
print(student)
```

```
(1, 'Rex', [85, 90, 95])
```

Unpack a list

In [54]:
```
lst = [10,25,75]
m1, m2, m3 = lst # brackets are optional
print(m1, m2, m3)
```

```
10 25 75
```

Modify stud_avg() to find average for the student tuple

In [35]:
```
def stud_avg2(student):

    #unpack it
    roll, name, marks = student

    total = 0
    for mark in marks:
        total += mark

    avg = total / len(marks)
    print("Roll No: ", roll, ' Name:', name, ' Average: ', avg)
```

In [37]:
```
# call it
student = (1, 'Rita', [85, 90, 95])
stud_avg2(student)
```

```
Roll No:  1  Name: Rita  Average:  90.0
```

In [38]:
```
# with many marks
stud2 = (2, 'Sweeta', [85, 90, 95, 80, 75])
stud_avg2(stud2)
```

```
Roll No:  2  Name: Sweeta  Average:  85.0
```

You can create list of students

In [45]:
```
students_list = [(1, 'Rita', [85, 90, 95]),
                 (2, 'Sweeta', [85, 90, 95, 80, 75]),
                 (3, 'Annette', [50, 60, 70, 80])]
```

In [46]:
```
print(students_list)
```

```
[(1, 'Rita', [85, 90, 95]), (2, 'Sweeta', [85, 90, 95, 80, 75]), (3, 'Annette',
[50, 60, 70, 80])]
```

Modify stud_avg2() to process the list of students

In [43]:
```
def stud_avg3(students):
    ''' student is the list of tuples'''

    for student in students:

        #unpack it
        roll, name, marks = student

        total = 0
        for mark in marks:
            total += mark

        avg = total / len(marks)
        print("Roll No: ", roll, ' Name:', name, ' Average: ', avg)

```

In [47]:
```
#call it
stud_avg3(students_list)
```

```
Roll No:  1  Name: Rita  Average:  90.0
Roll No:  2  Name: Sweeta  Average:  85.0
Roll No:  3  Name: Annette  Average:  65.0
```

Comparing tuples

In [50]:
```
(0, 1, 2) < (0, 3, 4)
```

Out[50]: True

In [51]:
```
(0, 10, 2) < (0, 3, 4)
```

Out[51]: False

In [52]:
```
(0, 1, 200000) < (0, 3, 4)
```

Out[52]: True

Walk Through: Print sorted list based on word length

text = 'hello c fine cpp'
Output (increasing order):
c
cpp
fine
hello

In [60]:

```
def sort_words(text):

    #split words
    words = txt.split()
    t = list()

    # append words with their length, t contains set of tuples
    for word in words:
        t.append((len(word), word))

    # sort set of tuples in increasing order of length
    #t.sort()

    # sort set of tuples in decreasing order
    t.sort(reverse=True)

    res = list()

    # unpack it
    for length, word in t:
        res.append(word)

    print(res)
```

In [61]:

```
text = 'but soft what light in yonder window breaks'
sort_words(text)
```

```
['yonder', 'window', 'breaks', 'light', 'what', 'soft', 'but', 'in']
```

Week - 7 List Processing

Write a function replace(s, old, new) that replaces all occurrences of old with new in a string s:

1. replace("Mississippi", "i", "I") as "MIssIssIppI" song = "I love spom! Spom is my favorite food. Spom, spom, yum!"
2. replace(song, "om", "am") as "I love spam! Spam is my favorite food. Spam, spam, yum!"
   3.replace(song, "o", "a") as 8 "I lave spam! Spam is my favarite faad. Spam, spam

In [1]:
```
def repl(string,old,new):
    for word in string.split():
        a=word.replace(old,new)
        print(a)
repl('Mississippi','i','I')
```

```
MIssIssIppI
```

In [2]:
```
song = "I love spom! Spom is my favorite food. Spom, spom, yum!"
repl(song, "om", "am")
```

```
I
love
spam!
Spam
is
my
favorite
food.
Spam,
spam,
yum!
```

2. [Weight management System] Write a weight loss program that prompts the user to enter in 7 days of weight values.

In [13]:

```
for i in range(7):
    c=float(input("enter ur weight"))
    li.append(c)
print("first day weight Programming Challenges – List Processing",li[0])
print("last day weight",li[-1])
print("highest weight is",max(li))
print("lowest weight is",min(li))
print("average of weight is",round(sum(li)/len(li)))
```

```
enter ur weight23.5
enter ur weight23.7
enter ur weight23.5
enter ur weight24.0
enter ur weight24.2
enter ur weight25.0
enter ur weight25.1
first day weight 23.5
last day weight 25.1
highest weight is 25.1
lowest weight is 23.5
average of weight is 24
```

3. [Store student's marks] Create a list that contains 5 marks as float values. Store the marks to a file remember to convert float to str before writing. Close the file. Then, open the file and print the average mark.

In [4]:

```
#2.

marks=[75.5,80.75,90.25,65.75]
def average(marks):
    return sum(marks)/len(marks)
a=average(marks)
with open('file.txt','w')as f:
    for s in marks:
        f.write(str(s)+'\n')
with open('file.txt','r')as f:
    marks=[line.strip('\n')for line in f]
    print(marks,end=" ")
print('Average:',a)

```

```
['75.5', '80.75', '90.25', '65.75'] Average: 78.0625
```

4. Python program to find N largest element from given list of integers, Function returns N largest elements

In [1]:
```
def Nmaxelements(list1, N):
    final_list = []

    for i in range(0, N):
        max1 = 0

        for j in range(len(list1)):
            if list1[j] > max1:
                max1 = list1[j];

        list1.remove(max1);
        final_list.append(max1)

    print(final_list)

# Driver code
list1 = [98,89,78,56,99,100,92]
N = 3

# Calling the function
Nmaxelements(list1, N)

```

```
[100, 99, 98]
```

5. Create a list in Python to store 10 items. Write a program to display the items at odd positions i.e.1, 3, 5, 7, ...

In [22]:
```
li=['apple','mango','orange','grapes','guava','dates','strawberry','cherry','
```

In [30]:
```
# iterate over string
for index in range(len(li)):
    # check if index is divisible by 2
    if index % 2 == 1:
        # print character at index
        print(li[index], end=',')
```

```
mango,grapes,dates,cherry,pista,
```

6. Create two lists in Python to store 10 items in each of them. Write a program to concatenate the lists.

In [2]:
```
li=['apple','mango','orange','grapes','guava','dates','strawberry','cherry','
li1=['carrot','potato','pumpkin','cabage','beetroot','beans','onion','tamrind
c=li+li1
print(c)
```

```
['apple', 'mango', 'orange', 'grapes', 'guava', 'dates', 'strawberry', 'cherr
y', 'nuts', 'pista', 'carrot', 'potato', 'pumpkin', 'cabage', 'beetroot', 'bean
s', 'onion', 'tamrind', 'lemon', 'brinjal']
```

7. Write a program in Python to create a list which contains n integers entered by the user

and delete all odd numbers from the list. Example: If initial contents of the list is= [1,2,3,4,5,6,7,8,9,10], the output would be=[2,4,6,8,10]

In [44]:
```
#.3.

# list with EVEN and ODD number
list = [1,2,3,4,5,6,7,8,9,10]
for i  in list:
    if(i%2 == 1):
            list.remove(i)

# print list after removing ODD elements
print ("list after removing ODD numbers:")
print (list)

```

```
list after removing ODD numbers:
[2, 4, 6, 8, 10]
```

8. Write a program in Python to create a list which contains n integers entered by the user and insert the sum of these values at the end of the list. Example: If initial contents of the list is=[1,2,3,4,5,6,7,8,9,10], the output would be= 1,2,3,4,5,6,7,8,9,10,55].

In [1]:
```
lst = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
sum = 0
for num in lst:
    sum += num
print(sum)
lst.append(sum)
print(lst)
```

```
55
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 55]
```

9. Write a program in Python to create a list which contains n integers entered by the user and add 2 to each of the even numbers in the list. Example: If initial contents of the list is= [1,2,3,4,5,6,7,8,9,10], the output would be=[1,4,3,6,5,8,7,10,9,12]

In [33]:
```
li = [1,2,3,4,5,6,7,8,9,10]
for i  in li:
    if(i%2 == 0):
        c=i+2
        print(c)
print (li)
```

```
4
6
8
10
12
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

10. Write a program in Python to create a list which contains n integers entered by the user and subsequently, check whether a given value exists in the list or not. Example: If initial

contents of the list is=[1,2,3,4,5,6,7,8,9,10] and the value to be searched is 10, then the output would be “Yes, 10 is present in the list!!!!”

In [20]:
```
lst = []
l=int(input("enter ur limit"))
for i in range(l):
    nu  = int(input("Enter the integer:"))
    lst.append(nu)
print(lst)
search=int(input("enter ur search number"))
if search in lst:
    print('Yes')
else:
    print('No')
```

```
enter ur limit5
Enter the integer:2
Enter the integer:3
Enter the integer:4
Enter the integer:5
Enter the integer:6
[2, 3, 4, 5, 6]
enter ur search number4
Yes
```

11. Write a Python program to extend a list without append function by inserting elements at any position.

In [3]:
```
my_list = [2,3,4,5,6,7,8]
my_list.extend('no')
print (my_list)
```

```
[2, 3, 4, 5, 6, 7, 8, 'n', 'o']
```

In [31]:
```
#another way
li=[8,9,0,4,5,3,5]
li.insert(2,'hai')
print(li)
```

```
[8, 9, 'hai', 0, 4, 5, 3, 5]
```

12. Write a Python program to generate and print a list except for the first 5 elements, where the values are square of numbers between 1 and 30 (both included) and generate all permutations of that list.

In [4]:
```
import itertools
def values():
    l=list()
    for i in range(1,31):
        l.append(i**2)
    print(l[5:])
    print("permuation is ")
    print(list(itertools.permutations([l])))
values()
```

```
[36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400, 441, 48
4, 529, 576, 625, 676, 729, 784, 841, 900]
permuation is
[([1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 32
4, 361, 400, 441, 484, 529, 576, 625, 676, 729, 784, 841, 900],)]
```

In [5]:
```
#3) Write a Python program to find the list in a list of lists whose sum of e
#Sample lists: [[1,2,3], [4,5,6], [10,11,12], [7,8,9]] Expected Output: [10,
list1 = [[1, 2, 3], [4, 5, 6], [10, 11, 12], [7, 8, 9]]
max(list1)
```

Out[5]:
```
[10, 11, 12]
```

13. Python program to find the list in a list of lists whose sum of elements is the highest using sum and max function

In [6]:
```
def maximumSum(list1):
    return(sum(max(list1, key = sum)))

list1 = [[1, 2, 3], [4, 5, 6], [10, 11, 12], [7, 8, 9]]
print (maximumSum(list1) )
print(max(list1))
```

```
33
[10, 11, 12]
```

14. Write a Python program to get a list, sorted in increasing order by the last element in each tuple from a given list of non-empty tuples and remove all duplicates from that list.

In [32]:
```
def last(n):
    return n[-1]
def sort(tuples):
    return sorted(tuples, key=last)

print(sort([(2, 5), (1, 2), (4, 4), (2, 5), (2, 1),(3,0)]))

```

```
[(3, 0), (2, 1), (1, 2), (4, 4), (2, 5), (2, 5)]
```

15. Write a function to remove duplicate elements in a list.

In [7]:
```
def Remove(duplicate):
    final_list = []
    for num in duplicate:
        if num not in final_list:
            final_list.append(num)
    return final_list


duplicate = [(2, 5), (1, 2), (4, 4), (2, 5), (2, 1),(3,0)]
print(Remove(duplicate))
```

```
[(2, 5), (1, 2), (4, 4), (2, 1), (3, 0)]
```

16. Write a Python program to count the elements in a list until an element is a tuple and find a tuple, the smallest second index value from that list. Python program to count the items until a list is encountered.

In [22]:
```
def Count(li):
    counter = 0
    for num in li:
        if isinstance(num, tuple):
            break
        counter = counter + 1
    return counter

li = [4, 5,(6, 10),  (1, 2, 3), 11, 2, 4]
print(Count(li))
```

```
2
```

17. (a) Write a Python function isEven, that accepts an integer as parameter and prints "Even" if the number is even, and "Odd" otherwise. Write a Python Program that takes a number from the user and calls isEven function to display if the entered number is Even or Odd. (b)Write a Python Program that accepts n numbers and calls isEven function to check if they are even or odd.

In [8]:
```
def find_Evenodd(num):

    if(num%2==0):
        print(num," Is an even")
    else:
        print(num," is an odd")

find_Evenodd(23)
```

```
23  is an odd
```

In [33]:
```
num=int(input("enter ur va"))
find_Evenodd(num)
```

```
enter ur va24
24  Is an even
```

18. Write a Python Program that accepts n numbers and calls isEven function to check if

they are even or odd. Give number of elements present in list.

In [9]:
```
li=[]
a=int(input("enter"))
for i in range(a):
    num=int(input("enter no"))
    li.append(num)
for num in li:
    find_Evenodd(num)
```

```
enter5
enter no45
enter no23
enter no23
enter no14
enter no56
45  is an odd
23  is an odd
23  is an odd
14  Is an even
56  Is an even
```

19. (a) Write a Python function isPerfect, that accepts an integer as parameter and prints "Perfect" if the number is perfect, and "Not Perfect" otherwise. Write a Python Program that takes an integer from the user and calls isPerfect function to display whether the entered number is Perfect. (b) Write a Python Program that accepts n integers and calls isPerfect function to check whether they are perfect.

In [11]:
```
def isperfect(n):
    sum = 0
    for x in range(1, n):
        if n % x == 0:
            sum += x
    return sum==n

print(isperfect(6))
```

```
True
```

In [12]:
```
n=int(input("enter your no"))
isperfect(n)
```

```
enter your no5
```

Out[12]:
```
False
```

In [82]:
```
def isperfect(n):
    sum = 0
    for x in range(1, n):
        if n % x == 0:
            sum += x
    return sum==n
```

In [84]:

```
li=[]
a=int(input("enter"))
for i in range(a):
    n=int(input("enter no"))
    li.append(n)
for n in li:
    print("output",isperfect(n))

```

```
enter2
enter no3
enter no9
output False
output False
```

Week - 8 Dictionary

1. create a dictionary called inventory and perform the following operations ¶

In [2]:
```
#1. Assume a dictionary: inventory = {"apple": 15, "banana": 35, "grape": 12}
inventory={
    "apple":15,
    "grapes":12,
    "banana":35

}
```

In [3]:
```
#a.Show all inventories
print(inventory)
#b.Show the quantities of banana
print(inventory.get('banana'))
#c.How many items in the dictionary?
len(inventory)
#d.Does grapes exist in the dictionary?
if "grapes" in inventory:
    print("yes")
else:
    print("No")

```

```
{'apple': 15, 'grapes': 12, 'banana': 35}
35
yes
```

In [4]:
```
#e.Does pears exists in the dictionary?. If so, return its quantity, otherwis
if "pears" in inventory.keys():
    print(inventory.get('pears'))
else:
    inventory["pears"] = 0
print(inventory)

```

```
{'apple': 15, 'grapes': 12, 'banana': 35, 'pears': 0}
```

In [5]:
```
#f.Print all sorted keys
sorted(inventory.keys())
```

Out[5]:
```
['apple', 'banana', 'grapes', 'pears']
```

In [6]:
```
#g.Delete pears from dictionary
del inventory['pears']
print(inventory)
```

```
{'apple': 15, 'grapes': 12, 'banana': 35}
```

2. Develop a function add_inventory perform the followiing operations

In [7]:
```
#h.Develop a function add_inventory(inventory, fruit, quantity) that increase
#Add 100 bananas. Add 200 apples. Display inventory dictionary
def add_inventory(inventory,fruit,quantity):
    inventory[fruit]=inventory.get(fruit,0)+quantity

```

In [8]:
```
new_inventory={}
add_inventory(new_inventory,'cherry',30)
print(new_inventory)
```

```
{'cherry': 30}
```

In [9]:
```
add_inventory(new_inventory,'banana',70)
print(new_inventory)
add_inventory(new_inventory,'mango',0)
print(new_inventory)

```

```
{'cherry': 30, 'banana': 70}
{'cherry': 30, 'banana': 70, 'mango': 0}
```

In [10]:
```
#Add 100 bananas.
add_inventory(new_inventory,'banana',100)
print(new_inventory)

```

```
{'cherry': 30, 'banana': 170, 'mango': 0}
```

In [11]:
```
#Add 200 apples.
add_inventory(new_inventory,'apple',200)
print(new_inventory)
```

```
{'cherry': 30, 'banana': 170, 'mango': 0, 'apple': 200}
```

In [12]:
```
#Display inventory dictionary
add_inventory(new_inventory,'apple',200)
print(new_inventory)
```

```
{'cherry': 30, 'banana': 170, 'mango': 0, 'apple': 400}
```

3. Develop a function reduce_inventory(inventory, fruit, quantity) that decreases the inventory of 'fruit'.

In [13]:
```
#i.Develop a function reduce_inventory(inventory, fruit, quantity) that decre
#Reduce 50 apples. Reduce 25 bananas. Display inventory dictionary
def reduce_inventory(inventory,fruit,quantity):
    inventory[fruit]=inventory.get(fruit,0)-quantity

```

In [14]:
```
new_inventory={"apple":300,"mango":87,"banana":320}
#Reduce 50 apples.
reduce_inventory(new_inventory,'apple',30)
print(new_inventory)
```

```
{'apple': 270, 'mango': 87, 'banana': 320}
```

In [15]:
```
#Reduce 25 bananas.
reduce_inventory(new_inventory,'banana',25)
#Display inventory dictionary
print(new_inventory)
```

```
{'apple': 270, 'mango': 87, 'banana': 295}
```

4. Store inventory dictionary onto a file, 'fruits_inventory.p"

In [16]:
```
#j.Store dictionary onto a file, 'fruits_inventory.p"
import pickle

def main():
    new_inventory={"apple":300,"mango":87,"banana":320}
    print("Original:", new_inventory)
# Write the dictionary to the pickle file
    file = open("fruits_inventory.p", "wb")
    pickle.dump(new_inventory, file)
    file.close()
# Load the data from the pickle file into a dictionary

    file = open("fruits_inventory.p","rb")
    dict2 = pickle.load(file)
    file.close()
    print("Loaded:", new_inventory)

main()
```

```
Original: {'apple': 300, 'mango': 87, 'banana': 320}
Loaded: {'apple': 300, 'mango': 87, 'banana': 320}
```

5. find out top 3 students and their grades from a file using pickle.

In [17]:
```
#2. Consider the following grades file of students.
#"resultswnames.txt"
#Open the file and load the names and grades to a dictionary, 'grades'. Sort
#according to descending order and display the top-3 students with their grad


#j. Store dictionary onto a file, 'fruits_inventory.p"

import pickle
def main():
    print("inventories are : ", new_inventory)

    f = open("fruits.p","wb")
    pickle.dump(new_inventory, f)
    f.close()

    f = open("fruits.p","rb")
    dic1 = pickle.load(f)
    f.close()

    print("loaded file:" ,dic1)

main()
```

```
inventories are :  {'apple': 270, 'mango': 87, 'banana': 295}
loaded file: {'apple': 270, 'mango': 87, 'banana': 295}
```

In [29]:
```
# Write a Python program to print all unique values in a dictionary.
L = [{"cs":"S001"}, {"ds": "S002"}, {"cs": "S001"}, {"ca": "S005"}, {"ds":"S0
print("Original List: ",L)
u_value = set( val for dic in L for val in dic.values())
print("Unique Values: ",u_value)

```

```
Original List:  [{'cs': 'S001'}, {'ds': 'S002'}, {'cs': 'S001'}, {'ca': 'S00
5'}, {'ds': 'S005'}, {'cs': 'S009'}, {'ds': 'S007'}]
Unique Values:  {'S002', 'S005', 'S007', 'S001', 'S009'}
```

6. use itertools and display the combinations of letters.

In [39]:
```
# Write a Python program to create and display all combinations of letters,
#selecting each letter from a different key in a dictionary.
import itertools
d ={'1':['a','b'], '2':['c','d']}
for combo in itertools.product(*[d[k] for k in sorted(d.keys())]):
    print(''.join(combo))

```

```
ac
ad
bc
bd
```

7. Using dictionary find the square value of its key.

In [43]:
```
#Write a Python program to print a dictionary where the keys are numbers betw
#and the values are square of keys and find the average value in a dictionary
dict={}
for x in range(1,17):
    dict[x]=x**2
print(dict)

```

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100, 11: 121,
12: 144, 13: 169, 14: 196, 15: 225, 16: 256}
```

In [45]:
```
def squarerDict(start, end):
    dict = {}
    for i in range(start, end+1):
        dict[i] = i ** 2
    return dict
print(squarerDict(1,16))
```

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100, 11: 121,
12: 144, 13: 169, 14: 196, 15: 225, 16: 256}
```

8.Using counter tool to combine values in python list of dictionaries.

In [46]:
```
#4) Write a Python program to combine values in python list of dictionaries.
#Sample data: [{'item': 'item1', 'amount': 400}, {'item': 'item2', 'amount':
    #Expected Output: Counter({'item1': 1150, 'item2': 300})
from collections import Counter
item_list = [{'item': 'item1', 'amount': 400}, {'item': 'item2', 'amount': 30
result = Counter()
for d in item_list:
    result[d['item']] += d['amount']
print(result)

```

```
Counter({'item1': 1150, 'item2': 300})
```

In [48]:
```
d=[{'item': 'item1', 'amount': 400}, {'item': 'item2', 'amount': 300}, {'item
d1={}
for i in d:
    if i['item'] in d1.keys():
        d1[i['item']]+=i['amount']
    else:
        d1[i['item']]=i['amount']
print(d1)
```

```
{'item1': 1150, 'item2': 300}
```

9. Consider the following grades file of students. "resultswnames.txt" Open the file and load the names and grades to a dictionary, 'grades'. Sort the dictionary according to descending order and display the top-3 students with their grades.

In [5]:
```
f = open("resultnames.txt",'w')
f.write("\nJohnny:8.65")
f.write("\nJuan:9.12")
f.write("\nJoseph:8.45")
f.write("\nStacey:7.81")
f.write("\nNideen:8.05")
f.write("\nZack:7.21")
f.write("\nAaron:8.31")
f.close()
f = open("resultnames.txt")
a = f.read()
print(a)
```

```
Johnny:8.65
Juan:9.12
Joseph:8.45
Stacey:7.81
Nideen:8.05
Zack:7.21
Aaron:8.31
```

In [61]:
```
grades = {"Johnny":8.65,
"Juan":9.12,
"Joseph":8.45,
"Stacey":7.81,
"Nideen":8.05,
"Zack":7.21,
"Aaron":8.31}
print(grades)
f = open("H:\\R.txt","w")
f.write(str(grades))
```

```
{'Johnny': 8.65, 'Juan': 9.12, 'Joseph': 8.45, 'Stacey': 7.81, 'Nideen': 8.05,
'Zack': 7.21, 'Aaron': 8.31}
```

Out[61]:
```
107
```

In [62]:
```
print(max(grades))
print(max(grades.values()))
```

```
Zack
9.12
```

In [63]:
```
sorteds = sorted(grades.keys(), reverse = True)
```

In [64]:
```
print(sorteds)
```

```
['Zack', 'Stacey', 'Nideen', 'Juan', 'Joseph', 'Johnny', 'Aaron']
```

In [67]:
```
1 sorteds1 = sorted(grades.values(), reverse = True)
```

In [68]:
```
1 print(sorteds1)
```

```
[9.12, 8.65, 8.45, 8.31, 8.05, 7.81, 7.21]
```

10. Write a program that asks the user to enter in a series of student names and test scores Use a Dictionary to store a new record in the dictionary based on the student name

In [72]:
```
1  marks = {}
2
3  while(True):
4      row = input("Enter the name and marks :")
5
6
7      info = row.split()
8
9      if(info[0] == "done"):
10         break
11
12     name  = info[0]
13     score = info[1]
14
15     marks[name] = score
16
17 # Print the contacts
18 print("\nPrinting name and scores:")
19 for name, score in marks.items():
20     print(name, ":", score)
```

```
Enter the name and marks :s 89
Enter the name and marks :g 56
Enter the name and marks :g 89
Enter the name and marks :done

Printing name and scores:
s : 89
g : 89
```

In [73]:
```
1 #@Extension: print them out in alphabetical order
2 sorted(marks.items())
```

Out[73]:
```
[('g', '89'), ('s', '89')]
```

11. Allow the user to enter in multiple scores for each student. If the student already has a score simply add it to the total score for the student. (Hint: use list to store several scores)

In [2]:

```
num = int(input("Please enter number of students:"))
student = {}
data = ['Mark 1 : ', 'Mark 2 : ', 'Mark 3 : ']
for i in range(0,num):
    name = input("Name :")
    student[name] = {}
    for i in data:
        student[name][i] = int(input(i))


# getting average mark of the student
print("Enter the student name to get the average marks")
name = input("Student name : ")
if name in student.keys():
    print ("Average marks : ", str(sum(student[name].values())/3.0))
else:
    print("please enter valid name")
```

```
Please enter number of students:2
Name :surya
Mark 1 : 90
Mark 2 : 98
Mark 3 : 85
Name :hari
Mark 1 : 97
Mark 2 : 89
Mark 3 : 87
Enter the student name to get the average marks
Student name : surya
Average marks :  91.0
```

In [3]:

```
name = input("Student name : ")
if name in student.keys():
    print ("Average marks : ", str(sum(student[name].values())/3.0))
else:
    print("please enter valid name")
```

```
Student name : surya
Average marks :  91.0
```

12. Read name and telephone numbers of customers until name==’done’. Store name and telephone number as key-value pairs of dictionary, ‘telephone’.

In [79]:

```
#

contacts = {}

while(True):
    row = input("Enter the name and number: ")

    info = row.split()

    if(info[0] == "done"):
        break

    name  = info[0]
    phone = info[1]

    contacts[name] = phone

print("\nPrinting Contacts:")
for name, phone in contacts.items():
    print(name, ":", phone)
```

```
Enter the name and number: R 9876543210
Enter the name and number: S 8907654321
Enter the name and number: done

Printing Contacts:
R : 9876543210
S : 8907654321
```

In [81]:

```python
def get_phone_number(phone_book, name):

    names = phone_book.keys()

    name_in_dict = name in names

    if name_in_dict:
        number = phone_book[name]
        return number
    else:
        return -1

def main():

    phone_book = {"R": 9876543210, "S" : 8907654321}

    print("Enter a name and we'll give you the phone number!")
    name = input("Name: ")

    number = get_phone_number(phone_book, name)

    number_exists = number != -1

    if number_exists:
        print("Number:", number)
    else:
        print("Sorry, the name", name, "doesn't exist in the phone book!")

main()
```

```
Enter a name and we'll give you the phone number!
Name: R
Number: 9876543210
```

In [82]:

```
#1. Implement a print_words(filename) function that counts how often each wor
#text and prints:

def word_count_dict(filename):

    word_count = {}

    file = open(filename, 'r')

    for line in file:
        words = line.split()

        for word in words:
            word = word.lower()


            # Special case if we're seeing this word for the first time.
        if not word in word_count:
            word_count[word] = 1
        else:
            word_count[word] = word_count[word] + 1
    file.close()
    return word_count
```

Week - 9 Regular Expression

In [8]:

```
# get file name
filename = input('Enter file name: ')
#open the file
try:
    f = open(filename)
except:
        print('File cannot be opened:', filename)
        exit()
lines = 0
for line in f:
    #if line.startswith("From "):
    words = line.split()
    print(words)
    lines+=1
print("There are ",lines," lines in the file with From as the first word")
```

```
Enter file name: E:\website.txt
['stephen.marquard@uct.ac.za']
['louis@media.berkeley.edu']
['zqian@umich.edu']
['rjlowe@iupui.edu']
['zqian@umich.edu']
['rjlowe@iupui.edu']
['cwen@iupui.edu']
['cwen@iupui.edu']
['gsilver@umich.edu']
['gsilver@umich.edu']
['zqian@umich.edu']
['gsilver@umich.edu']
['wagnermr@iupui.edu']
['zqian@umich.edu']
['antranig@caret.cam.ac.uk']
['gopal.ramasammycook@gmail.com']
['david.horwitz@uct.ac.za']
['david.horwitz@uct.ac.za']
['david.horwitz@uct.ac.za']
['david.horwitz@uct.ac.za']
['stephen.marquard@uct.ac.za']
['louis@media.berkeley.edu']
['louis@media.berkeley.edu']
['ray@media.berkeley.edu']
['cwen@iupui.edu']
['cwen@iupui.edu']
['cwen@iupui.edu']
There are  27  lines in the file with From as the first word
```

Week - 10 Object Oriented Analysis and Design

Class and Object

Template for a real world thing. It can abstract states and behavious of things

In [1]:
```
class Apple(object):

    # constructor initializes instance variables and
    # are automatically called, when object is created
    def __init__(self):
        self.color = 'red'

    def show(self):
        print('Color: ', self.color, 'Message: Apple a day keeps the doctor a

# create and call method
a = Apple()
a.show()
```

```
Color:  red Message: Apple a day keeps the doctor away
```

Constructor can have arguments

In [2]:
```
class Apple(object):

    def __init__(self, n):   # initialize with constructor
        self.name = n
        self.color = 'red'

    def show(self):
        print('Name: ', self.name, ' Color: ', self.color)

```

In [3]:
```
# create 2 objects and provide values to constructor
royal = Apple("Royal")
kashmir = Apple('Kashmir')

# call instance methods with dot
royal.show()
kashmir.show()
```

```
Name:  Royal  Color:  red
Name:  Kashmir  Color:  red
```

1. Create a class Student with instance variables rollno, name, class.

Use init() to initialize them. Define a method display() to print their details

Object Life Cycle - constructor and destructor

In [4]:
```
class Bank(object):
    total_branches = 0

    def __init__(self):
        print('Bank object is constructed')

    def open_branch(self, city) :
        self.branch = city
        Bank.total_branches += 1
        print('Branch opened at ', self.branch)
        print('Total branches opened so far is ', Bank.total_branches)

    def __del__(self):
        print('Bank object is destructed')
```

Song class

In [5]:
```
class Song(object):

    def __init__(self, lyric):
        self.lyrics = lyric

    def singSong(self):
        for line in self.lyrics:
            print(line)

happy_birthday = Song(['Happy birthday to you', 'Happy birthday to you',
                       'Happy birthday to dear RK', 'Happy birthday to you\n']

london_bridges = Song(['London Bridge is falling down', 'Falling down, fallin
                       'London Bridge is falling down', 'My fair lady\n'])

happy_birthday.singSong()
london_bridges.singSong()
```

```
Happy birthday to you
Happy birthday to you
Happy birthday to dear RK
Happy birthday to you

London Bridge is falling down
Falling down, falling down
London Bridge is falling down
My fair lady
```

1. Create 2 more songs and sing them

2. Put the lyrics in a separate variable, then pass that variable to the class to use instead

3. Add one more method and call it

Instances can be arguments of functions

In [6]:
```
# create Apple class
class Apple(object):

    # constructor is automatically called, when object is created
    def __init__(self, c):
        self.color = c

    def taste(self):
        return self.color + ' apple tastes good'
```

In [7]:
```
# method receives Apple instance
def eat(apple):
    print(apple.taste())

# instantiate Apple class
apple1 = Apple('red')

# call eat()
eat(apple1)
```

```
red apple tastes good
```

Instances can return values

In [8]:
```
# method returns Apple instance
def getApple():
    apple = Apple('green')
    return apple

# receive an apple object
apple2 = getApple()
eat(apple2)
```

```
green apple tastes good
```

Walkthrouth: BankAccount class

In [9]:
```
class BankAccount(object):

    def __init__(self, name1):
        self.balance = 0.0
        self.name = name1

    def credit(self, amt):
        self.balance += amt
        print(self.name, ' account : ', amt, ' is credited.', ' Current balan

    def debit(self, amt):
        self.balance -= amt
        print(self.name, ' account : ', amt, ' is debited.', ' Current balanc

```

In [10]:
```
# create bank accounts
a1 = BankAccount('John')
a2 = BankAccount('Rita')

a1.credit(10000)
a1.debit(500)
a1.credit(2000)

# Similarly perform transactions for 'Rita
```

```
John  account :  10000  is credited.  Current balance:  10000.0
John  account :  500  is debited.  Current balance:  9500.0
John  account :  2000  is credited.  Current balance:  11500.0
```

Walk Through

1: Remove print statements from credit() and debit() methods and define a new method show_balance() that displays the current balance

2: Modify the above code by adding instance variable accountno in init(). Include values for accountno while creating objects

3: Create a class Customer with instance variables accountno, name and address inside init().

Class variables

Class variables belong to class and not instances. So all instances can access the same class variable. It is also called static variables

In [11]:

```
# Simple example

class Dog(object):
    kind = 'Raja Palayam'

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

# instantiate Dog
d1 = Dog('Tommy')
d2 = Dog('Shinny')

print(d1.name)
print(d2.name)

print(d1.kind)
print(d2.kind)
```

```
Tommy
Shinny
Raja Palayam
Raja Palayam
```

Another Example

In [12]:

```
class Transaction(object):
    #class variable
    trans_count = 0

    def transact(self):
        Transaction.trans_count += 1
        print(' Total no of transaction happend in our bank so far is ', Tran

```

In [13]:

```
# instantiate
john = Transaction()
rita = Transaction()

# John performs transaction using ATM, bank or online banking
john.transact()
rita.transact()
```

```
Total no of transaction happend in our bank so far is  1
Total no of transaction happend in our bank so far is  2
```

In [14]:
```
# instantiate Bank class
sbi = Bank()
sbi.open_branch('Trichy')

# create more branches
# sbi = None  # it destructs object sbi

# create another bank pnb
# open branches for pnb
# discard pnb object
```

```
Bank object is constructed
Branch opened at  Trichy
Total branches opened so far is  1
```

Walk Through: Practice more transact()

Class Methods and Static Methods

In [15]:
```
class Bank(object):

    rbi_deposit = 100

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

    @classmethod
    def get_approval(cls, amt):
        approved = False
        if amt == Bank.rbi_deposit:
            approved = True
        return approved

    @staticmethod
    def is_license_valid(years):
        return years<10


    def open_branch(self, city): ## 100 crore for approval

        amount = 100

        if Bank.get_approval(amount):
            self.branch = city
            self.total_branches += 1
            print('For ', self.name, ', branch opened at ', self.branch)
            print('Total branches opened far ', self.name , ' is ', self.tota
        else:
            print('RBI did not approve new branch for ', self.name)

```

In [16]:

```
# instantiate Bank
pnb = Bank('PNB')
pnb.open_branch('Trichy')
pnb.open_branch('Chennai')
pnb.open_branch('Covai')
print('For ', pnb.name, ' license valid is ', Bank.is_license_valid(9))

sbi = Bank('SBI')
sbi.open_branch('Trichy')
sbi.open_branch('Chennai')
print('For ', sbi.name, ' license valid is ', Bank.is_license_valid(14))
```

```
For  PNB , branch opened at  Trichy
Total branches opened far  PNB  is  1
For  PNB , branch opened at  Chennai
Total branches opened far  PNB  is  2
For  PNB , branch opened at  Covai
Total branches opened far  PNB  is  3
For  PNB  license valid is  True
Bank object is destructed
For  SBI , branch opened at  Trichy
Total branches opened far  SBI  is  1
For  SBI , branch opened at  Chennai
Total branches opened far  SBI  is  2
For  SBI  license valid is  False
```

Inheritance

Class accessing data and functions of parent class. It defines IS-A relationship

In [17]:

```
class Transaction(object):
    #class variable
    trans_count = 0

    def transact(self):
        print('Transacted in the bank branch')
        Transaction.trans_count += 1
        print(' Total no of transaction happend in our bank so far is ', Tran

class ATMTransaction(Transaction):

    def card_transact(self):
        print('Transacted using ATM card')
        Transaction.trans_count += 1
        print('Total no of transaction happend in our bank so far is ', Trans

t1 = ATMTransaction()
t1.card_transact()
t1.transact()

t2 = ATMTransaction()
t2.card_transact()
t2.transact()
```

```
Transacted using ATM card
Total no of transaction happend in our bank so far is  1
Transacted in the bank branch
 Total no of transaction happend in our bank so far is  2
Transacted using ATM card
Total no of transaction happend in our bank so far is  3
Transacted in the bank branch
 Total no of transaction happend in our bank so far is  4
```

2. Create a class Fruit that initializes instance variable size=10. It defines a method taste() that returns a string 'so sweet'. Create a child class Apple that defines a method getcolor() that returns a color, say 'red'. Instantiate Apple class and create an object 'a'. With instance 'a', call both methods and also print the value of the variable size

In [18]:
```
class Fruit(object):

    def __init__(self):
        self.size = 10

    def taste(self):
        return 'So sweet'

class Apple(Fruit):
    def getcolor(self):
        return 'red'

# create object for Apple
a = Apple()

# call methods
print(a.getcolor())
print(a.taste())

# you can call base class's variable also
print(a.size)
```

```
red
So sweet
10
```

In [ ]:
```

```

Overriding

Child class defining a method as same as parent class's method. It is a way of achieving Polymorphism

In [19]:

```
## create Parent class with a method override()
class Parent(object):

    def override(self):
        print('Inside Parent override() function')

## create child class with the same name override() but different body
class Child(Parent):

    def override(self):
        print('Inside Child override() function')

# instantiate parent class and call its method
dad = Parent()
dad.override()

# instantiate child class and call its method
son = Child()
son.override()
```

```
Inside Parent override() function
Inside Child override() function
```

Call base class's overridden method using super()

In [20]:

```
class Parent(object):

    def override(self):
        print('Inside Parent override() function')

class Child(Parent):

    def override(self):
        print('Inside Child override() function')
        super(Child, self).override()

dad = Parent()
dad.override()

son = Child()
son.override()

```

```
Inside Parent override() function
Inside Child override() function
Inside Parent override() function
```

Multiple inheritance

Child class can inherit two or more Parent classes

In [21]:

```
class Fruit(object):
    def color(self):
        print('Every Fruit has color')

class Food(object):
    def energy(self):
        print('Every Food has energy')

class Apple(Fruit, Food):  # inherits from Fruit and Food
    def healthy(self):
        print('Apple is healthy')

a = Apple()
a.color()
a.energy()
a.healthy()
```

```
Every Fruit has color
Every Food has energy
Apple is healthy
```

Another Example

In [22]:

```
class Father(object):

    def __init__(self):
        self.fname = 'Rex'

    def show(self):
        print('I am father ', self.fname)

class Mother(object):

    def __init__(self):
        self.mname = 'Rita'

    def show(self):
        print('I am mother ', self.mname)

class Child(Father, Mother):

    def __init__(self):
        Father.__init__(self) # explicitly call parent
        Mother.__init__(self) # here too
        self.myname = 'Robin'

    def show(self):
        print('I am child. My name is ', self.myname)
        print('My parents are ', self.fname, ' and ', self.mname)

rex = Father()
rita = Mother()
robin = Child()

rex.show()
rita.show()
robin.show()

```

```
I am father  Rex
I am mother  Rita
I am child. My name is  Robin
My parents are  Rex  and  Rita
```

Multilevel Inheritance

Child class inherits from Parent class which inturn inherits from Grand Parent class. Python supports any number of levels

In [23]:

```
class Food(object):
    def energy(self):
        print('Every Food has energy')

class Fruit(Food):
    def color(self):
        print('Every Fruit has color')

class Apple(Fruit):  # inherits from Fruit and Food
    def healthy(self):
        print('Apple is healthy')

#instantiate Apple
royalapple = Apple();
royalapple.healthy()
royalapple.color()
royalapple.energy()

```

```
Apple is healthy
Every Fruit has color
Every Food has energy
```

Week - 11 Retrieving Data from Web

Retrieving web pages using HTTP protocol

In [1]:
```
1  import socket
2
3  mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
4
5  mysock.connect(('data.pr4e.org', 80))
6
7  cmd = 'GET http://data.pr4e.org/romeo.txt HTTP/1.0\r\n\r\n'.encode()
8
9  mysock.send(cmd)
10
11 while True:
12     data = mysock.recv(512)
13
14     if len(data) < 1:
15         break
16
17     print(data.decode(),end='')
18
19 mysock.close()
```

```
HTTP/1.1 200 OK
Date: Sat, 31 Aug 2019 01:18:03 GMT
Server: Apache/2.4.18 (Ubuntu)
Last-Modified: Sat, 13 May 2017 11:22:22 GMT
ETag: "a7-54f6609245537"
Accept-Ranges: bytes
Content-Length: 167
Cache-Control: max-age=0, no-cache, no-store, must-revalidate
Pragma: no-cache
Expires: Wed, 11 Jan 1984 05:00:00 GMT
Connection: close
Content-Type: text/plain

But soft what light through yonder window breaks
It is the east and Juliet is the sun
Arise fair sun and kill the envious moon
Who is already sick and pale with grief
```

Retrieve Images using HTTP

In [5]:

```
import socket
import time

HOST = 'data.pr4e.org'
PORT = 80
mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
mysock.connect((HOST, PORT))
mysock.sendall(b'GET http://data.pr4e.org/cover3.jpg HTTP/1.0\r\n\r\n')

count = 0
picture = b""

while True:
    data = mysock.recv(5120)
    if len(data) < 1:
        break

    #time.sleep(0.25)
    count = count + len(data)
    print(len(data), count)
    picture = picture + data

mysock.close()

# Look for the end of the header (2 CRLF)
pos = picture.find(b"\r\n\r\n")
print('Header length', pos)
print(picture[:pos].decode())

# Skip past the header and save the picture data
picture = picture[pos+4:]
fhand = open("stuff.jpg", "wb")
fhand.write(picture)
fhand.close()

# you can display stuff.jpg as well
from IPython.display import Image
display(Image(filename='stuff.jpg'))
```

```
5120 5120
5120 10240
4280 14520
5120 19640
5120 24760
5120 29880
5120 35000
5120 40120
536 40656
5120 45776
2140 47916
5120 53036
5120 58156
5120 63276
612 63888
1452 65340
5120 70460
```

```
5044 75504
```

Retrieving web pages using urllib

In [6]:
```
1  import urllib.request
2
3  fhand = urllib.request.urlopen('http://data.pr4e.org/romeo.txt')
4  for line in fhand:
5      print(line.decode().strip())
```

```
But soft what light through yonder window breaks
It is the east and Juliet is the sun
Arise fair sun and kill the envious moon
Who is already sick and pale with grief
```

Print word frequency of a web page

In [7]:
```
1  import urllib.request, urllib.parse, urllib.error
2
3  fhand = urllib.request.urlopen('http://data.pr4e.org/romeo.txt')
4
5  counts = dict()
6  for line in fhand:
7      words = line.decode().split()
8      for word in words:
9          counts[word] = counts.get(word, 0) + 1
10 print(counts)
```

```
{'But': 1, 'soft': 1, 'what': 1, 'light': 1, 'through': 1, 'yonder': 1, 'windo
w': 1, 'breaks': 1, 'It': 1, 'is': 3, 'the': 3, 'east': 1, 'and': 3, 'Juliet':
1, 'sun': 2, 'Arise': 1, 'fair': 1, 'kill': 1, 'envious': 1, 'moon': 1, 'Who':
1, 'already': 1, 'sick': 1, 'pale': 1, 'with': 1, 'grief': 1}
```

Walk-Through

Use urllib to retrieve the document from a URL, display up to 3000 characters, and count the overall number of characters in the document. Don't worry about the headers for this exercise, simply show the first 3000 characters of the document contents.

Retrieve Images and save it locally

In [11]:

```
import urllib.request, urllib.parse, urllib.error

img = urllib.request.urlopen('http://data.pr4e.org/cover3.jpg').read()
fhand = open('cover3.jpg', 'wb')
fhand.write(img)
fhand.close()

# show the save file too
from IPython.display import Image
#display(Image(filename='cover3.jpg'))
```

Retrive images block by block to avoid system crash

In [10]:

```
import urllib.request, urllib.parse, urllib.error

img = urllib.request.urlopen('http://data.pr4e.org/cover3.jpg')
fhand = open('cover3.jpg', 'wb')
size = 0
while True:
    info = img.read(100000)
    if len(info) < 1:
        break
    size = size + len(info)
    fhand.write(info)

print(size, 'characters copied.')
fhand.close()
```

```
230210 characters copied.
```

Parsing HTML using regular expressions

In [12]:

```python
# Search for link values within URL input
import urllib.request, urllib.parse, urllib.error
import re
import ssl

# Ignore SSL certificate errors
ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE

url = input('Enter website:> ')
html = urllib.request.urlopen(url).read()
links = re.findall(b'href="(http[s]?://.*?)"', html)
for link in links:
    print(link.decode())

```

```
Enter website:> https://docs.python.org (https://docs.python.org)
https://docs.python.org/3/index.html (https://docs.python.org/3/index.html)
https://www.python.org/ (https://www.python.org/)
https://docs.python.org/3.9/ (https://docs.python.org/3.9/)
https://docs.python.org/3.8/ (https://docs.python.org/3.8/)
https://docs.python.org/3.7/ (https://docs.python.org/3.7/)
https://docs.python.org/3.6/ (https://docs.python.org/3.6/)
https://docs.python.org/3.5/ (https://docs.python.org/3.5/)
https://docs.python.org/2.7/ (https://docs.python.org/2.7/)
https://www.python.org/doc/versions/ (https://www.python.org/doc/versions/)
https://www.python.org/dev/peps/ (https://www.python.org/dev/peps/)
https://wiki.python.org/moin/BeginnersGuide (https://wiki.python.org/moin/Begin
nersGuide)
https://wiki.python.org/moin/PythonBooks (https://wiki.python.org/moin/PythonBo
oks)
https://www.python.org/doc/av/ (https://www.python.org/doc/av/)
https://www.python.org/ (https://www.python.org/)
https://www.python.org/psf/donations/ (https://www.python.org/psf/donations/)
http://sphinx.pocoo.org/ (http://sphinx.pocoo.org/)
```

Parsing HTML using BeautifulSoup

In [13]:

```
import urllib.request, urllib.parse, urllib.error
from bs4 import BeautifulSoup
import ssl

# Ignore SSL certificate errors
ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE

url = input('Enter website:> ')
html = urllib.request.urlopen(url, context=ctx).read()
soup = BeautifulSoup(html, 'html.parser')

# Retrieve all of the anchor tags
tags = soup('a')
for tag in tags:
    print(tag.get('href', None))

```

```
Enter website:> https://www.python.org (https://www.python.org)
#content
#python-network
/
/psf-landing/
https://docs.python.org (https://docs.python.org)
https://pypi.python.org/ (https://pypi.python.org/)
/jobs/
/community/
#top
/
/psf/donations/
#site-map
#
javascript:;
javascript:;
javascript:;
#
https://www.facebook.com/pythonlang?fref=ts (https://www.facebook.com/pythonl
```

Walk-Through

program to extract and count para-graph (p) tags from the retrieved HTML document and display the count of the paragraphs as the output of your program. Do not display the paragraph text, only count them. Test your program on several small web pages as well as some larger web pages.

Retrieve all anchor tags

In [14]:

```
from urllib.request import urlopen
from bs4 import BeautifulSoup
import ssl

# Ignore SSL certificate errors
ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE

url = input('Enter :> ')
html = urlopen(url, context=ctx).read()
soup = BeautifulSoup(html, "html.parser")

# Retrieve all of the anchor tags
tags = soup('a')
for tag in tags:
    # Look at the parts of a tag
    print('TAG:', tag)
    print('URL:', tag.get('href', None))
    print('Contents:', tag.contents[0])
    print('Attrs:', tag.attrs)

```

```
Enter - http://www.dr-chuck.com/page1.htm (http://www.dr-chuck.com/page1.htm)
TAG: <a href="http://www.dr-chuck.com/page2.htm">
Second Page</a>
URL: http://www.dr-chuck.com/page2.htm (http://www.dr-chuck.com/page2.htm)
Contents:
Second Page
Attrs: {'href': 'http://www.dr-chuck.com/page2.htm'}
```

XML Data Processing

Extract xml elements

In [15]:

```
import xml.etree.ElementTree as ET

data = '''
<person>
  <name>Chuck</name>
  <phone type="intl">
    +1 734 303 4456
  </phone>
  <email hide="yes" />
</person>'''

tree = ET.fromstring(data)
print('Name:', tree.find('name').text)
print('Attr:', tree.find('email').get('hide'))
```

```
Name: Chuck
Attr: yes
```

Looping xml data

In [16]:

```
import xml.etree.ElementTree as ET

input = '''
<stuff>
  <users>
    <user x="2">
      <id>001</id>
      <name>Chuck</name>
    </user>
    <user x="7">
      <id>009</id>
      <name>Brent</name>
    </user>
  </users>
</stuff>'''

stuff = ET.fromstring(input)
lst = stuff.findall('users/user')  #include parent level elements 'users'
print('User count:', len(lst))

for item in lst:
    print('Name', item.find('name').text)
    print('Id', item.find('id').text)
    print('Attribute', item.get('x'))
```

```
User count: 2
Name Chuck
Id 001
Attribute 2
Name Brent
Id 009
Attribute 7
```

JavaScript Object Notation(JSON) Data Processing

In [17]:

```
import json

data = '''
[
  { "id" : "001",
    "x" : "2",
    "name" : "Chuck"
  } ,
  { "id" : "009",
    "x" : "7",
    "name" : "Brent"
  }
]'''

info = json.loads(data)
print('User count:', len(info))

for item in info:
    print('Name', item['name'])
    print('Id', item['id'])
    print('Attribute', item['x'])
```

```
User count: 2
Name Chuck
Id 001
Attribute 2
Name Brent
Id 009
Attribute 7
```

Retrieve Twitter user's timeline data

In [1]:

```
import urllib.request, urllib.parse, urllib.error
import twurl
import ssl

# https://apps.twitter.com/
# Create App and get the four strings, put them in hidden.py

TWITTER_URL = 'https://api.twitter.com/1.1/statuses/user_timeline.json'

# Ignore SSL certificate errors
ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE

while True:
    print('')
    acct = input('Enter Twitter Account:')
    if (len(acct) < 1): break
    url = twurl.augment(TWITTER_URL,
                        {'screen_name': acct, 'count': '2'})
    print('Retrieving', url)
    connection = urllib.request.urlopen(url, context=ctx)
    data = connection.read().decode()
    print(data[:250])
    headers = dict(connection.getheaders())
    # print headers
    print('Remaining', headers['x-rate-limit-remaining'])
```

```
Enter Twitter Account:drsris
Retrieving https://api.twitter.com/1.1/statuses/user_timeline.json?oauth_consum
er_key=ObG0rFIZON5mHUcXHbwhBIB5A&oauth_timestamp=1567238775&oauth_nonce=6767581
9&oauth_version=1.0&screen_name=drsris&count=2&oauth_token=836931359880835072-s
rBegAOc9dC3oWWoJFKrvlBzng8Trk6&oauth_signature_method=HMAC-SHA1&oauth_signature
=%2BG8CGqI62Sq4nALY4n3ceoCzxog%3D (https://api.twitter.com/1.1/statuses/user_ti
meline.json?oauth_consumer_key=ObG0rFIZON5mHUcXHbwhBIB5A&oauth_timestamp=156723
8775&oauth_nonce=67675819&oauth_version=1.0&screen_name=drsris&count=2&oauth_to
ken=836931359880835072-srBegAOc9dC3oWWoJFKrvlBzng8Trk6&oauth_signature_method=H
MAC-SHA1&oauth_signature=%2BG8CGqI62Sq4nALY4n3ceoCzxog%3D)
[{"created_at":"Wed Aug 28 11:28:42 +0000 2019","id":1166673959112380416,"id_st
r":"1166673959112380416","text":"@Wikipedia is close to my heart. Most of what
 I know is from #Wikipedia \nToday I was happy to give a small donation\u2026 h
ttps:\/\/t.co\ (https:\/\/t.co\)
Remaining 899

Enter Twitter Account:
```

Retrieve friends of Twitter user

In [2]:

```
import urllib.request, urllib.parse, urllib.error
import twurl
import json
import ssl

# https://apps.twitter.com/
# Create App and get the four strings, put them in hidden.py

TWITTER_URL = 'https://api.twitter.com/1.1/friends/list.json'

# Ignore SSL certificate errors
ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE

while True:
    print('')
    acct = input('Enter Twitter Account: ')
    if (len(acct) < 1): break
    url = twurl.augment(TWITTER_URL,
                        {'screen_name': acct, 'count': '5'})
    print('Retrieving', url)
    connection = urllib.request.urlopen(url, context=ctx)
    data = connection.read().decode()

    js = json.loads(data)
    print(json.dumps(js, indent=2))

    headers = dict(connection.getheaders())
    print('Remaining', headers['x-rate-limit-remaining'])

    for u in js['users']:
        print(u['screen_name'])
        if 'status' not in u:
            print('   * No status found')
            continue
        s = u['status']['text']
        print('  ', s[:50])
```

```
Enter Twitter Account: drsris
Retrieving https://api.twitter.com/1.1/friends/list.json?oauth_consumer_key=O
bG0rFIZON5mHUcXHbwhBIB5A&oauth_timestamp=1567238795&oauth_nonce=18658553&oaut
h_version=1.0&screen_name=drsris&count=5&oauth_token=836931359880835072-srBeg
AOc9dC3oWWoJFKrvlBzng8Trk6&oauth_signature_method=HMAC-SHA1&oauth_signature=I
2KhNzlWVCg4lDUYj%2BJnL32n8ps%3D (https://api.twitter.com/1.1/friends/list.jso
n?oauth_consumer_key=ObG0rFIZON5mHUcXHbwhBIB5A&oauth_timestamp=1567238795&oau
th_nonce=18658553&oauth_version=1.0&screen_name=drsris&count=5&oauth_token=83
6931359880835072-srBegAOc9dC3oWWoJFKrvlBzng8Trk6&oauth_signature_method=HMAC-
SHA1&oauth_signature=I2KhNzlWVCg4lDUYj%2BJnL32n8ps%3D)
{
  "users": [
    {
      "id": 918068370246852608,
      "id_str": "918068370246852608",
      "name": "suneet gupta",
      "screen_name": "suneet4037",
```

```
"location": "",
```

Week - 12 Google Geolocation API Access

Extraction location information using Google Geocoding API

In [1]:

```
import urllib.request, urllib.parse, urllib.error
import json
import ssl

api_key = False
# If you have a Google Places API key, enter it here
# api_key = 'AIzaSy___IDByT70'
# https://developers.google.com/maps/documentation/geocoding/intro

if api_key is False:
    api_key = 42
    serviceurl = 'http://py4e-data.dr-chuck.net/json?'
else :
    serviceurl = 'https://maps.googleapis.com/maps/api/geocode/json?'

# Ignore SSL certificate errors
ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE

while True:

    address = input('Enter location: ')

    if len(address) < 1:
        break

    parms = dict()
    parms['address'] = address

    if api_key is not False:
        parms['key'] = api_key

    url = serviceurl + urllib.parse.urlencode(parms)

    print('Retrieving', url)
    uh = urllib.request.urlopen(url, context=ctx)

    data = uh.read().decode()
    print('Retrieved', len(data), 'characters')

    try:
        js = json.loads(data)
    except:
        js = None

    if not js or 'status' not in js or js['status'] != 'OK':
        print('==== Failure To Retrieve ====')
        print(data)
        continue

    print(json.dumps(js, indent=4))

    lat = js['results'][0]['geometry']['location']['lat']
    lng = js['results'][0]['geometry']['location']['lng']
    print('lat', lat, 'lng', lng)
```

```
57
58      location = js['results'][0]['formatted_address']
59      print(location)
60
```

```
Enter location: trichy
Retrieving http://py4e-data.dr-chuck.net/json?address=trichy&key=42 (http://py4e-data.dr-chuck.net/json?address=trichy&key=42)
Retrieved 1769 characters
{
    "results": [
        {
            "address_components": [
                {
                    "long_name": "Tiruchirappalli",
                    "short_name": "Tiruchirappalli",
                    "types": [
                        "locality",
                        "political"
                    ]
                },
                {
                    "long_name": "Tiruchirappalli",
                    "short_name": "Tiruchirappalli",
```

In [ ]:

```
1
```

Week - 13 Using Databases and SQL

Create a database and table

In [1]:
```
import sqlite3

conn = sqlite3.connect('music.sqlite')
cur = conn.cursor()

cur.execute('DROP TABLE IF EXISTS Tracks')
cur.execute('CREATE TABLE Tracks (title TEXT, plays INTEGER)')
```

Out[1]:
```
<sqlite3.Cursor at 0x182734ba6c0>
```

Insert two rows of values

In [2]:
```
cur.execute('INSERT INTO Tracks (title, plays) VALUES (?, ?)',
    ('Thunderstruck', 20))
cur.execute('INSERT INTO Tracks (title, plays) VALUES (?, ?)',
    ('My Way', 15))
conn.commit()
```

Show the tows

In [3]:
```
print('Tracks:')
cur.execute('SELECT title, plays FROM Tracks')
for row in cur:
     print(row)

```

```
Tracks:
('Thunderstruck', 20)
('My Way', 15)
```

Delete all rows where plays < 18

In [4]:
```
cur.execute('DELETE FROM Tracks WHERE plays < 18')
conn.commit()

cur.execute('SELECT * FROM Tracks')
for row in cur:
    print(row)

cur.close()
```

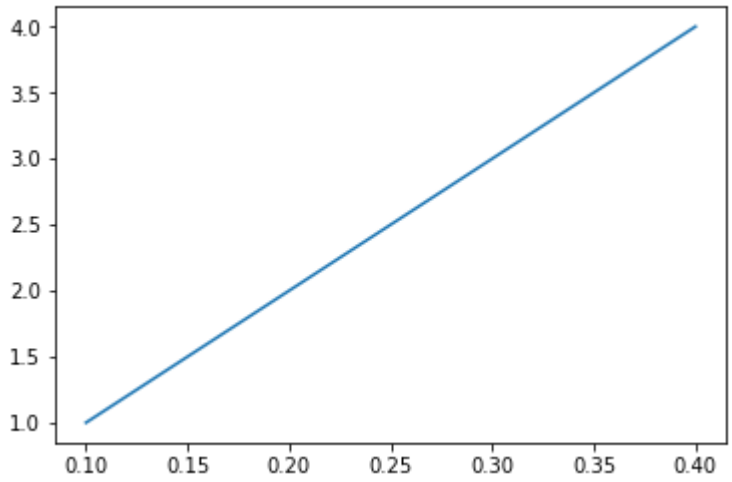
```
('Thunderstruck', 20)
```

Week - 14 Plotting Graphs using matplotlib

1. Line graphs

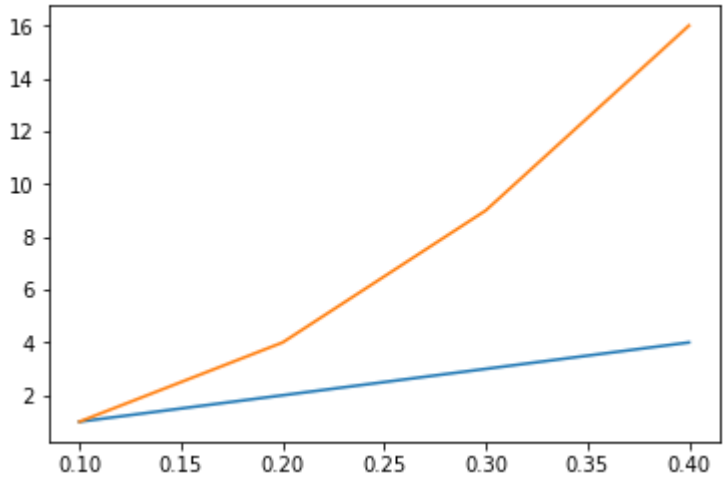
In [1]:
```
1  from matplotlib import pyplot as plt
2  %matplotlib inline
3
4  # inputs [x] and [y]
5  plt.plot([0.1, 0.2, 0.3, 0.4], [1, 2, 3, 4])
```

Out[1]:
```
[<matplotlib.lines.Line2D at 0x2d649c065c0>]
```



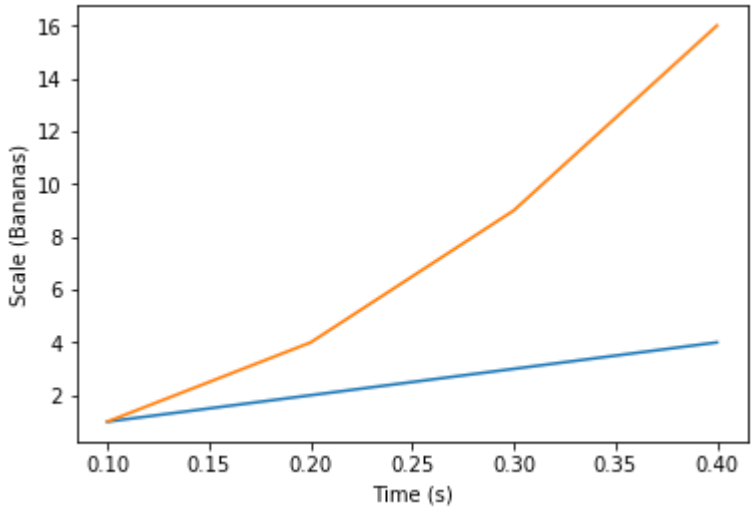
In [2]:
```
1  # plot multiple curves
2  plt.plot([0.1, 0.2, 0.3, 0.4], [1, 2, 3, 4],
3  [0.1, 0.2, 0.3, 0.4], [1, 4, 9, 16])
```

Out[2]:
```
[<matplotlib.lines.Line2D at 0x2d649cac7f0>,
 <matplotlib.lines.Line2D at 0x2d649cac940>]
```



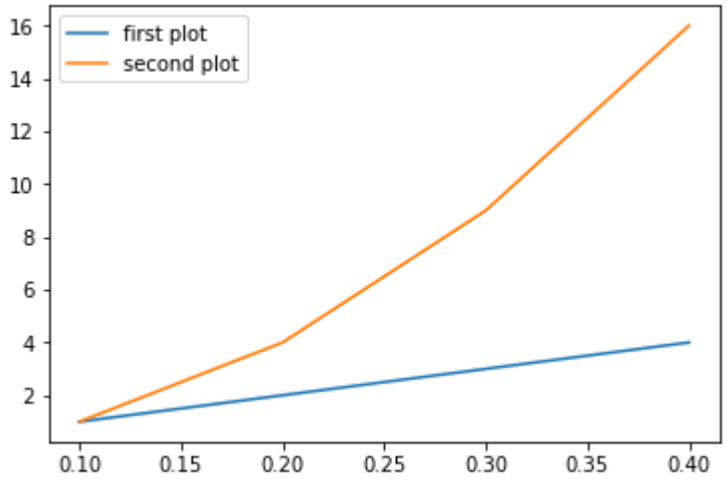
In [3]:
```
# Labels for x and y axis
plt.plot([0.1, 0.2, 0.3, 0.4], [1, 2, 3, 4])
plt.plot([0.1, 0.2, 0.3, 0.4], [1, 4, 9, 16])
plt.xlabel("Time (s)")
plt.ylabel("Scale (Bananas)")
```

Out[3]:
```
Text(0, 0.5, 'Scale (Bananas)')
```



In [4]:
```
# add legends for each curve
plt.plot([0.1, 0.2, 0.3, 0.4], [1, 2, 3, 4], label='first plot')
plt.plot([0.1, 0.2, 0.3, 0.4], [1, 4, 9, 16], label='second plot')
plt.legend()
```

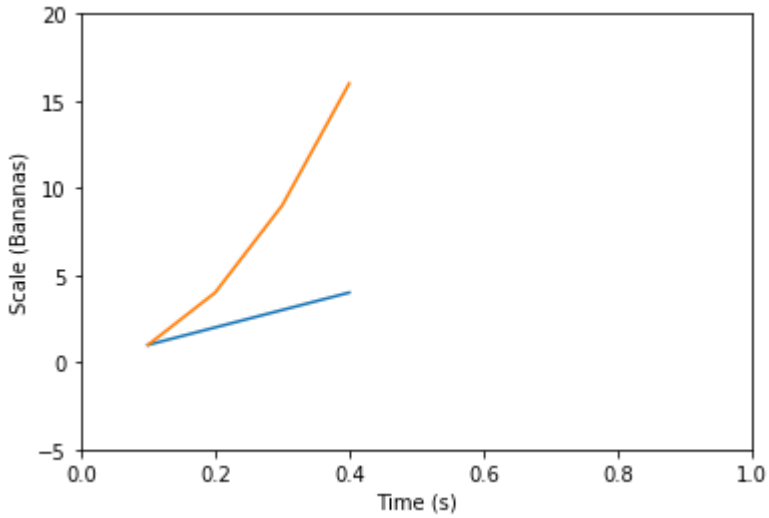
Out[4]:
```
<matplotlib.legend.Legend at 0x2d649d8b2b0>
```



In [5]:

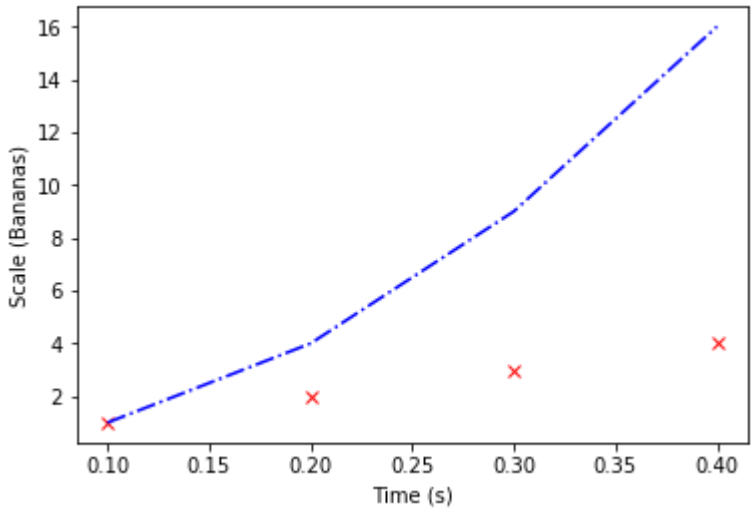
```
# you can set axis range
plt.plot([0.1, 0.2, 0.3, 0.4], [1, 2, 3, 4])
plt.plot([0.1, 0.2, 0.3, 0.4], [1, 4, 9, 16])
plt.xlabel("Time (s)")
plt.ylabel("Scale (Bananas)")
plt.xlim(0, 1)
plt.ylim(-5, 20)
```

Out[5]: (-5, 20)



In [6]:
```
1  # plotting styles
2  plt.plot([0.1, 0.2, 0.3, 0.4], [1, 2, 3, 4], 'rx')
3  plt.plot([0.1, 0.2, 0.3, 0.4], [1, 4, 9, 16], 'b-.')
4  plt.xlabel("Time (s)")
5  plt.ylabel("Scale (Bananas)")
```

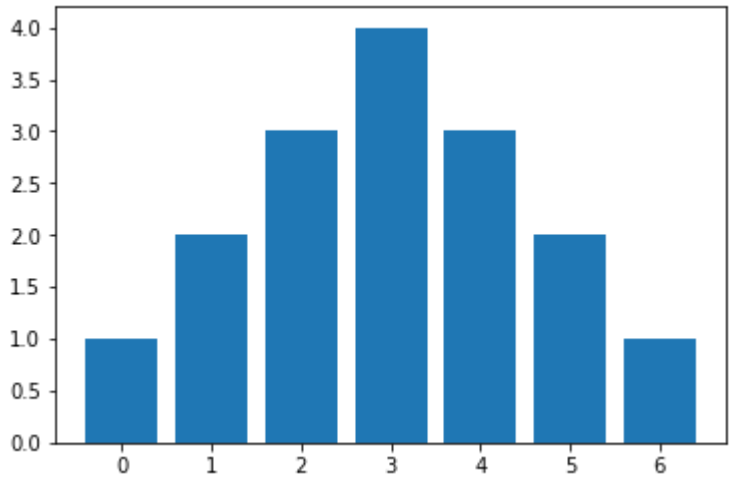
Out[6]:
```
Text(0, 0.5, 'Scale (Bananas)')
```



2. Bar chart

In [7]:
```
1  plt.bar(range(7), [1, 2, 3, 4, 3, 2, 1])
```

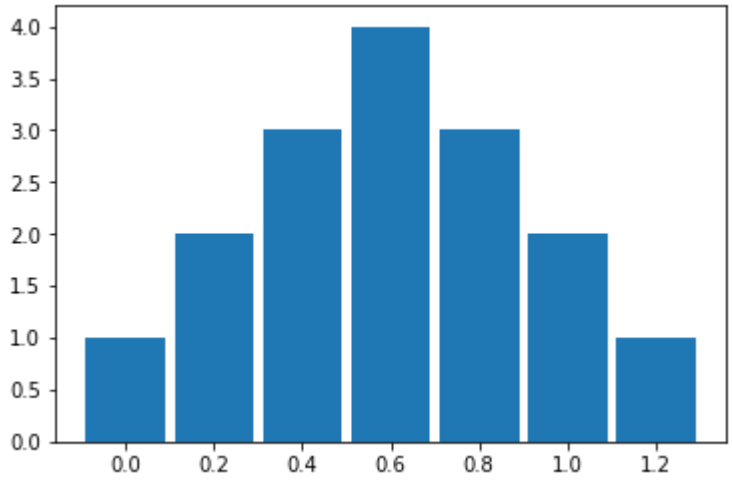
Out[7]:
```
<BarContainer object of 7 artists>
```



With range for x axis and width of bar

In [11]:
```
1 import numpy
2 plt.bar(numpy.arange(0., 1.4, .2), [1, 2, 3, 4, 3, 2, 1], width=0.18)
```

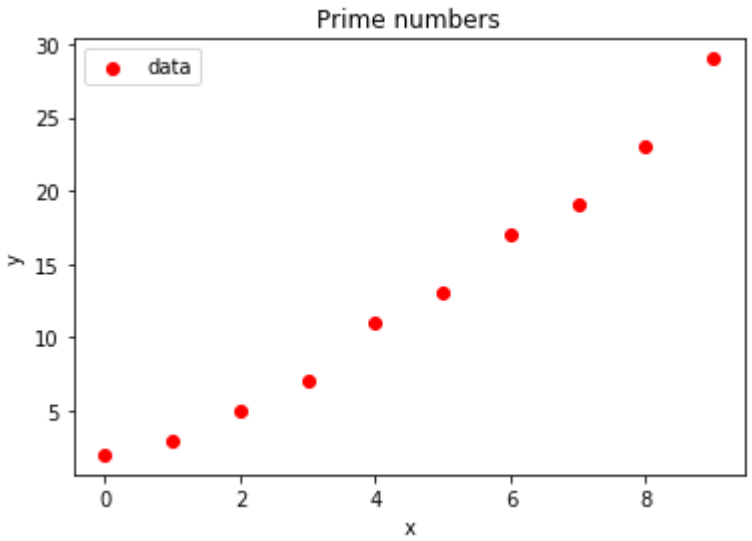
Out[11]: <BarContainer object of 7 artists>



3. Scatter plot

In [13]:

```
x_data = list(range(10))
y_data = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29]

plt.scatter(x_data, y_data, c='r', label='data')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Prime numbers')
plt.legend()
plt.show()
```

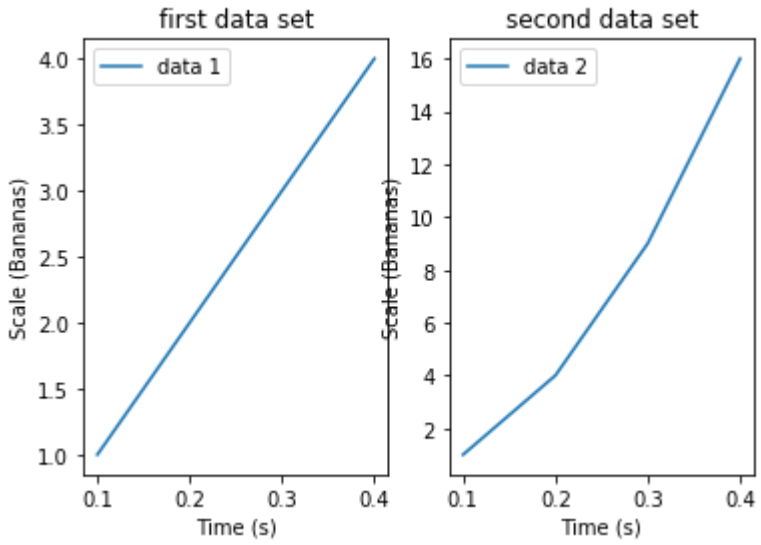


You can save plots using savefig() method
plt.savefig(filename)

Plot multiple graphs

In [14]:

```
x_data1 = [0.1, 0.2, 0.3, 0.4]
y_data1 = [1, 2, 3, 4]

x_data2 = [0.1, 0.2, 0.3, 0.4]
y_data2 = [1, 4, 9, 16]

fig = plt.figure()
ax1 = fig.add_subplot(1, 2, 1)
ax2 = fig.add_subplot(1, 2, 2)

ax1.plot(x_data1, y_data1, label='data 1')
ax2.plot(x_data2, y_data2, label='data 2')

ax1.set_xlabel('Time (s)')
ax1.set_ylabel('Scale (Bananas)')
ax1.set_title('first data set')
ax1.legend()

ax2.set_xlabel('Time (s)')
ax2.set_ylabel('Scale (Bananas)')
ax2.set_title('second data set')
ax2.legend()

plt.show()
```



Week - 15 Simple Website Design using Flask

In [ ]:

```python
#app.py
from flask import Flask
from flask import render_template

app = Flask(__name__)

@app.route("/")
@app.route('/hello')

def main():
    return 'Hello world'
    #return render_template("first_app.html")

@app.route('/hello/<username>')  # URL with a variable
def hello_username(username):    # The function shall take the URL variable a
    return 'Hello, {}'.format(username)


@app.route('/hello/<int:userid>')  # Variable with type filter. Accept only i
def hello_userid(userid):          # The function shall take the URL variable
    return 'Hello, your ID is: {:d}'.format(userid)


if __name__ == "__main__":
    app.run()

```

In [ ]:

```html
#first_app.html
<html>
<head>
  <title>My First App</title>
</head>
<body>
<div> Hi, this is my first Flask Web Application</div>
</body>
</html>
```

In [ ]:

```html
#j2_query.html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Entry Page</title>
</head>
<body>
<form action="process" method="post">
  <label for="username">Please enter your name: </label>
  <input type="text" id="username" name="username"><br>
  <input type="submit" value="SEND">
</form>
</body>
</html>
```

In [ ]:

```html
#js_response.html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Response Page</title>
</head>
<body>
  <h1>Hello, {{ username }}</h1>
</body>
</html>
```

In [ ]:

```python
#hello_form.py
from flask import Flask, render_template, request
app = Flask(__name__)

@app.route('/')
def main():
    return render_template('j2_query.html')

@app.route('/process', methods=['POST'])
def process():
    # Retrieve the HTTP POST request parameter value from 'request.form' dict
    _username = request.form.get('username')  # get(attr) returns None if att

    # Validate and send response
    if _username:
        return render_template('j2_response.html', username=_username)
    else:
        return 'Please go back and enter your name...', 400  # 400 Bad Reques

if __name__ == '__main__':
    app.run()
```