FOR LOOP :-

In [ ]:

1-Write a Python program to print numbers from 1 to 10 using a for loop.

In [125…

```
for number in range(11):
    print(number)
```

```
0
1
2
3
4
5
6
7
8
9
10
```

- we use the for loop to print the numbers from to 10.
- we use the print statements to print the numbers .

2-Explain the difference between a for loop and a while loop in Python.

`Difference b/w for loop and while loop` :- Looping is nothing but its a concept by which you will be able to iterate data one by one and then own deta set write with own logic. for loop is used to iterate over a sequence of items but while loop is used to repeatedly excutute the block of statements while a condition is true .

- if you know how times you will iterate go with for.
- if you don't know how times you will iterate go with while loop

In [22]:

```
## Example of for loop.


items=["pen","book","shose","lunch"]

for items in items:
    print(items)
```

```
pen
book
shose
lunch
```

In [23]:

```
## Example of while loop .

items=["pen","book","shose","lunch"]

length_items=len(items)
index=0

while index <length_items:
    print(items[index])
    index+=1
```

```
pen
book
shose
lunch
```

3- Write a Python program to calculate the sum of all numbers from 1 to 100 using a for loop.

In [24]:

```
sum=0
for i in range(1,101):
    sum=sum+i
print("Sum of numbers is :",sum)
```

```
Sum of numbers is : 5050
```

- we create the variable sum to store the value of sum of numbers from 1 to 100.
- we use the for loop to calculate the sum of all numbers from 1 to 100.
- we use the print statements to print the sum of numbers .

4-How do you iterate through a list using a for loop in Python?

In [26]:

```
my_list=["avneesh verma","aniket verma","rahul verma","ravi verma","rajat

for i in my_list:
    print(i)
```

```
avneesh verma
aniket verma
rahul verma
ravi verma
rajat verma
```

- in the python with the help of for loop we iterate the data one by one unless and until complete data is not acess ,we explaine this things using the example of for loop,where in this example i is a varible where store the data one by one .

5-Write a Python program to find the product of all elements in a list using a for loop.

In [27]:

```
my_list=[1,2,3,4,5,6,7,8,9,10]
sum=0
for i in my_list:
    sum=sum+i
print("product of all elements in a list is :",sum)
```

```
product of all elements in a list is : 55
```

- first of all we create the list of numbers that we want to product of all elements.
- we create the variable variable name sum where i can store the data of product of all elements in a list.
- we use the print statements to print the product of all elements in a list.

6-Create a Python program that prints all even numbers from 1 to 20 using a for loop.

In [28]:

```
print("Even numbers b/w 1 to 10 is:")
for i in range(11):
    if i%2==0:
        print(i)
```

```
Even numbers b/w 1 to 10 is:
0
2
4
6
8
10
```

- We use the for loop to iterate a even numbers from 1 to 10.
- we use the if statements inside the for loop to iterate the the only even numbers from to the for loop.

7-Write a Python program that calculates the factorial of a number using a for loop.

In [30]:

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

print("Factorial of number is:",sum)
```

```
Factorial of number is: 3628800
```

- factorial of a number is calculated by multiplying it with all the numbers below it string from 1.
- we we the for loop to iterate a numbers from (1 ,n+1) and inside the for loop `sum=i*sum` :use this condition to find the factorial of a number .
- we use the print statements to print the factpoorial of the number .

8-How can you iterate through the characters of a string using a for loop in Python?

In [31]:

```
characters="pwskills"

print("iterate the characters:")
for i in characters:
    print(i)
```

```
iterate the characters:
p
w
s
k
i
l
```

```
i
l
s
```

- we create the characters string .
- we use the for loop to iterate the characters from the string .
- we use the print statements to print the characters.

-9 Write a Python program to find the largest number in a list using a for loop.

In [32]:

```
## create the list
my_list=[1,2,3,4,5,6,4,7,85]

## create the variable to store the largest value .
largest=my_list[0]

## we use the for loop to check the largest one.
for i in my_list:
    if i>largest:
        largest=i
print("Largest inside my_list is:",largest)
```

```
Largest inside my_list is: 85
```

- we create the list over here that we want to check largest one inside this.
- initilize the largest variable with the first element in the list.
- use the for loop to travere every element in the list and check the curent element i is grater than the value presenmt in the largest variable.
- If it is, then assign the value in i to the variable largest.
- After the loop ends, largest will store the largest element in the list.

10-Create a Python program that prints the Fibonacci sequence up to a specified limit using a for loop.

In [34]:

```
n=int(input("how meny term user is find"))
## create a two variable .
a=0
b=1
 ##  use the for loop
for i in range(0,n):
    print(a,end=" ")
    c=a+b
    a=b
    b=c
```

```
0 1 1 2 3 5 8 13 21 34
```

11-Write a Python program to count the number of vowels in a given string using a for loop.

In [35]:

```
string = input("enter the string")

count =0
for i in string:
        if (i=="a" or i=="e" or i=="i" or i=="o" or i== "u" or i=="A" or
            count = count+1
print("Number of vowels in this string is :",count)
```

```
Number of vowels in this string is : 5
```

- we ask the user input string that user want to check .

- we create the count variable to store the how many vowels present inside the string .
- we use the for loop to iterate each char through the for loop .
- inside the for loop we create the condition .
- we use the print statements to print the number of vowels.

12-Create a Python program that generates a multiplication table for a given number using a for loop.

In [36]:

```
## ask the user input

n=int(input("enter the nmber that you want to multiplication table"))
print("Table is :")

## use the for loop to genrate the table
for i in range(1,11):
    print(n*i)
```

```
Table is :
10
20
30
40
50
60
70
80
90
100
```

- we ask the user input that he want to genrate the table .
- we use the range(1,11) function to genrate the value of table til 10 numbers .
- `print(n*i)` : we use this function to genrate the perticular table that user want .

13-Write a Python program to reverse a list using a for loop.

In [37]:

```
my_list=["avneesh","verma","rahul","Ravi"]
new_list=[]
for i in range(len(my_list)):
    new_list.insert(i,my_list[-1])
    my_list.pop(-1)
print(new_list)
```

```
['Ravi', 'rahul', 'verma', 'avneesh']
```

14-Write a Python program to find the common elements between two lists using a for loop.

In [38]:

```
list1=[1,2,3,4,5,6,7,8,9]
list2=[1,2,10,11,12,13,14]
def common_element(list1,list2):
    result=[]
    for element in list1:
        if element in list2:
            result.append(element)

    return result
print(common_element(list1,list2))
```

```
[1, 2]
```

```
[1, 2]
```

- first of all we create the two list name list1 and list2 seprately .
- we define the function ,function name is common_element.
- we create the result variable to store the common element b/w list1 and list2.
- we use the for loop to iterate each element to list1.
- imside the for loop we use this `if element in list2:` : condition to check the common element b/w them.
- we append the value of elements inside the result variable.
- we use the print statements to print the common elements .

15-Explain how to use a for loop to iterate through the keys and values of a dictionary in Python.

In [39]:

```
my_dic={"name":"avneesh","mobile":142562}
for key in my_dic.keys():
    print("Key:",key)

for value in my_dic.values():
    print("Value:",value)
for key,values in my_dic.items():
    print("key:",key, '\t', "value:",values)
```

```
Key: name
Key: mobile
Value: avneesh
Value: 142562
key: name        value: avneesh
key: mobile      value: 142562
```

Explanation :-

- we create the dictionary ,dictionary name my_dict that contains some key and values .
- if i want to iterate the key at that point of time inside the for loop i used `my_dic.keys()` : , if i want to iterae value at that point of time i used the my_dic.values(): , and if i want to iterate key and value both at that point i used my_dic.items(): to iterate both key and values.

16-Write a Python program to find the GCD (Greatest Common Divisor) of two numbers using a for loop.

In [40]:

```
num1=60
num2=36
gcd=1
for i in range(1,max(num1,num2)):
    if num1%i==0 and num2%i==0:
        gcd = i
print("GCD of num1 & num 2 is :",gcd)
```

```
GCD of num1 & num 2 is : 12
```

- we create the two number variable num1 and num2 that store the data that we want to find the GCd of two numbers .
- we initilize the value of GCD is 1 .
- Run a loop in a iteration of b/w `[(1,max(num1,num2)]`
- note down that highest value divides both num1 and num2 .
- If i satisfies (num1 % i == 0 and num2 % i == 0) then new value of GCD is i.
- print the GCD of two numbers using print statements.

17-Create a Python program that checks if a string is a palindrome using a for loop.

In [43]:

```
string =input("enter your string ")

for i in string:
    if string == string[::-1]:
        print("string is a palindrom")

        break
else:
    print("string is not a palindrom")
```

```
string is a palindrom
```

- we ask the user input that he want to check string is palindrom or not .
- According this question i'll use the for loop ,inside this loop create a one condition if input string is equal to the reverse of the same string `if string == string[::-1]:` : then it condition will be true then print string is a palindrom .
- if this condition fails then else block comes into a picture then print string is not a palindrom.

18-Write a Python program to remove duplicates from a list using a for loop.

In [44]:

```
my_list=[1,2,3,4,1,23,44,2,3,4,926]
print("Before duplicate remove list:",my_list)
remove_dupl=[]

for i in my_list:
    if i not in remove_dupl:
        remove_dupl.append(i)
print("After duplicate remove list :",remove_dupl)
```

```
Before duplicate remove list: [1, 2, 3, 4, 1, 23, 44, 2, 3, 4, 926]
After duplicate remove list : [1, 2, 3, 4, 23, 44, 926]
```

- we create list ,list name is my_list that containse some inteser data.
- we print the before removing duplicate list and then print after removing duplicate list.
- we create the remove_dupl variable to store the after removing duplicate list .
- we use the for loop to remove the duplicate from the list inside the for loop we create the one condition `if i not in remove_dupl` : to check the duplicate inside the list.
- `remove_dupl.append(i)` : we use this function to append the data inside the remove_dupl variable.

19-Create a Python program that counts the number of words in a sentence using a for loop.

In [45]:

```
sentence = "Pw skils lab"
words = 0
for i in sentence:
    if i == " " :
        words += 1
if len(sentence) > 0:
    print("Number of words:", words + 1)
```

```
Number of words: 3
```

20-Write a Python program to find the sum of all odd numbers from 1 to 50 using a for loop.

In [46]:

```
## find the sum odd .
sum_odd =0
for i in range(1,50):
    if i%2 != 0:

        sum_odd= sum_odd+i
print(sum_odd)
```

```
625
```

- we create the sum_odd variable to store the sum of odd numbers from 1 to 50.
- we use the for loop method to find the sum of odd numbers , in this function we use the range function to sum of numbers from 1 to 50 .
- inside the for loop we use the if statements ,to find the odd numbers b/w 1 to 50 numbers .
- we `sum_odd= sum_odd+i` : use this function to sum of odd numbers .
- we use the print statements to print the odd numbers.

21-Write a Python program that checks if a given year is a leap year using a for loop.

In [48]:

```
start_year = int(input("Please select a starting year: "))
num_of_years = int(input("Please select how many years you'd like to chec

end_year =start_year+num_of_years

## use the for loop

for year in range(start_year-1, end_year):
    year += 1
    if year % 4 == 0 and (year % 100 != 0 or year % 400 == 0):
        print (year," Is a leap year")
    else:
        print (year)
```

```
2023
2024  Is a leap year
2025
2026
2027
2028  Is a leap year
```

- we ask the users enter the starting year & how many year you would like to check.
- `end_year =start_year+num_of_years` : we use this statements to count the end_year.
- we use the for loop to check the year is leap year or not .

22-Create a Python program that calculates the square root of a number using a for loop.

```
    if number < 0:
        return "Cannot calculate the square root of a negative number."

    # Initial guess
    guess = number / 2

    # Iterate for 10 iterations (adjust as needed)
    for _ in range(10):
        guess = 0.5 * (guess + number / guess)

    return guess

# Example usage:
if __name__ == "__main__":
    try:
        input_number = float(input("Enter a number to calculate its squar
        result = calculate_square_root(input_number)
        print(f"The square root of {input_number} is approximately {resul
    except ValueError:
        print("Invalid input. Please enter a valid number.")
```

```
The square root of 50.0 is approximately 13.500000
```

23-Write a Python program to find the LCM (Least Common Multiple) of two numbers using a for loop.

In [53]:

```
num1 = 12
num2 = 14
for i in range(max(num1, num2), 1 + (num1 * num2)):
    if i % num1 == i % num2 == 0:
        lcm = i
        break
print("LCM of", num1, "and", num2, "is", lcm)
```

```
LCM of 12 and 14 is 84
```

If else :-

1-Write a Python program to check if a number is positive, negative, or zero using an if-else statement.

In [56]:

```
number =int(input("enter number"))

if number <0:
    print("given number is negative")

elif number>0:
    print("given number is positive")
else:
    print("given number is zero")
```

```
given number is positive
```

- we ask the user input .
- we use the multiple if ,elif and else statements to check the number is possitive ,negative or zero.

2-Create a Python program that checks if a given number is even or odd using an if-else statement.

In [57]:

```
number = int(input("enter number"))
```

```
if number%2==0:
    print("given number is even:",number)
else:
    print("given number is odd:",number)
```

```
given number is even: 4
```

- we ask the user input that he want to check number is even or odd2.
- we use the if and else statements to check the number is even or odd .

3-How can you use nested if-else statements in Python, and provide an example?

Explanation :-

- if we use the if,else statements inside the if,else statements that is something called the nested if-else statements.

EXAMPLE :-

In [58]:

```
# Simpler example of a nested if-else statement
num = 10

if num > 0:
    if num % 2 == 0:
        print("The number is positive and even.")
    else:
        print("The number is positive but odd.")
else:
    print("The number is not positive.")
```

```
The number is positive and even.
```

4-Write a Python program to determine the largest of three numbers using if-else.

In [59]:

```
num1=52
num2=54
num3=10

if (num1>=num2) and (num1>=num3):
    gratest = num1
elif (num2>=num3) and (num2>=num1):
    gratest = num2

else:
    gratest = num3

print("The gratest number b/w them is:",gratest)
```

```
The gratest number b/w them is: 54
```

- we store the numbers in variable num1 , num2 and num3 .
- `if (num1>=num2) and (num1>=num3):` : this condition will be true then gratest number is num1.
- `elif (num2>=num3) and (num2>=num1):` : if this condition will be true then num2 is a gratest numbers. , else num3 is a gratest number.

5-Write a Python program that calculates the absolute value of a number using if-else.

In [60]:

```
number = int(input("enter the number"))

if number<0:
    print("Absulute value of a number is:",-number)

else:
    print("Number is absulute:",number)
```

```
Number is absulute: 5
```

- we ask the user input that he want to check the absulute value of a number.
- we use the if ,else statements to find the absulute value of any number given by the user.

6-Create a Python program that checks if a given character is a vowel or consonant using if-else.

In [61]:

```
vowels=["a","e","i","o","u","A","E","I","O","U"]

cha = input("enter the character")

if cha in vowels:
    print(cha,": is a vowels character")

else:
    print(cha,": is a consonant character")
```

```
10 : is a consonant character
```

- we create the vowels list in tha form of small and capital characters.
- we use the if and else statements to check the char is a vowels or consonant.

7-Write a Python program to determine if a user is eligible to vote based on their age using if-else.

In [62]:

```
eligible_vote_age=18
user_age=int(input("enter your age"))

if user_age <eligible_vote_age:
    print("you are not eligible for vote b/c your age is less than 18 yea

elif user_age>=18:
    print("You are eligible for vote ")

else:
    print("You enter the wrong age please try again")
```

```
you are not eligible for vote b/c your age is less than 18 year
```

- we ask the user age from the user.
- we `if user_age <eligible_vote_age:` : this condition to check the user is less than 18 or not if user is less than 18 then this condition true and then print "you are not eligible for vote b/c your age is less than 18 year"
- `elif user_age>=18:` : if this condition is true then print You are eligible for

8-Create a Python program that calculates the discount amount based on the purchase amount using if-else.

In [63]:

```
amount=int(input("Shopkeaper tell me price of paint and shirt"))

if 5000>=amount>2000:
    discount=amount*0.2

elif 7000>=amount>5000:
    discount=amount*0.3

elif 10000>=amount>7000:
    discount=amount*0.4
print("Discount on this prise is :",discount)
print("Net pay amount is:",amount-discount)
```

```
Discount on this prise is : 1000.0
Net pay amount is: 4000.0
```

- we ask the shopekeeper price of paint and shirt .
- we use the multiple if elif statements to decide the perticular discount on perticular price rate.
- we use the print statements to print the discount rate and net pay amount .

9-Write a Python program to check if a number is within a specified range using if-else.

In [64]:

```
uper_bound=100
lower_bound=1
number=int(input("enter the number b/w 1 to 100"))

if lower_bound<=number<=uper_bound:
    print("number is within in a specified range")

else:
    print("Number is out of specified range")
```

```
number is within in a specified range
```

- we decide the upper bound and lower bound .
- we ask the user input that he wanr

10-Create a Python program that determines the grade of a student based on their score using if-else.

In [65]:

```
score = int(input("Enter your score"))

if score >=90:
    print("Excelent your grade is A+")

elif score>=80:
    print("Excelent your grade is A")

elif score>=70:
    print("Excelent your grade is B+")
elif score>=60:
    print("Excelent your grade is B")

elif score>=50:
    print("Excelent your grade is C+")

else:
```

- we ask the user scoring marks .
- we use the multiple if , elif and else statements to decide the student grade .
- we use the print statements to print the grades.

11-Write a Python program to check if a string is empty or not using if-else.

In [66]:

```
def String(myString):
    if myString.strip():
        print("it's not an empty or blank string")
    else:
        print("it's an empty or blank string")
print(String(""))
```

```
it's an empty or blank string
None
```

In [67]:

```
def String(myString):
    if myString.strip():
        print("it's not an empty or blank string")
    else:
        print("it's an empty or blank string")
print(String("Pwskills"))
```

```
it's not an empty or blank string
None
```

12-Create a Python program that identifies the type of a triangle (e.g., equilateral, isosceles, or scalene) based on input values using if-else.

In [68]:

```
side1=int(input("enter the value of side1"))
side2=int(input("enter the value of side2"))
side3=int(input("enter the value of side3"))

if side1==side2==side3:
    print("This is a equilateral trangle")

elif side1==side2 !=side3 or side1==side3 !=side2 or side2==side3!=side1:
    print("This is a  isosceles trangle")

else:
    print("This is a scalene trangle")
```

```
This is a scalene trangle
```

- we ask the users angle sides ,side1 ,side2 and side3.
- if all the sides are equal than it is a equilateral trange.
- if any two sides are equal then it is a isosceles .
- if any of these sides not equal to then it is a scalene angle.

13-Write a Python program to determine the day of the week based on a user-provided number using if-else.

In [69]:

```
print(" 1-Sunday 2-Monday 3-tuesday 3-thursday 4-Friday 5-saterday")

day =int(input("enter the numbers"))
```

```
if day==1:
    print("Sunday")

elif day==2:
    print("Monday")

elif day==3:
    print("tuesday")

elif day==4:
    print("thursday")

elif day==5:
    print("saterday")

else:
    print("You enterd wrong input plese try again")
```

```
 1-Sunday 2-Monday 3-tuesday 3-thursday 4-Friday 5-saterday
Sunday
```

- we use the multiple if elif and else statements to print the day based on a user provide number.
- we ask the user number to print print the day.

14-Create a Python program that checks if a given year is a leap year using both if-else and a function.

In [71]:

```
n=int(input("enter the year"))
def leap_year(year):
    if year % 4 == 0 and (year % 100 != 0 or year % 400 == 0):
        print("year is a leap year")

    else:
        print("year is not a leap year")

print(leap_year(n))
```

```
year is not a leap year
None
```

- we define the function ,function name is leap_year .
- inside the function we use the if and else statements to check the given year is a leap year or not.

15-How do you use the "assert" statement in Python to add debugging checks within if-else blocks?

In [52]:

```
def divide_numbers(a, b):
    assert b != 0, "Division by zero is not allowed."

    result = a / b
    return result

numerator = 10
denominator = 0

if denominator != 0:
    result = divide_numbers(numerator, denominator)
    print(f"Result of division: {result}")
else:
    print("Denominator must be non-zero.")
```

```
Denominator must be non-zero.
```

```
def divide_numbers(a, b):
    assert b != 0, "Division by zero is not allowed."

    result = a / b
    return result
\
numerator = 10
denominator = 0

if denominator != 0:
    result = divide_numbers(numerator, denominator)
    print(f"Result of division: {result}")
else:
    print("Denominator must be non-zero.")
```

```
Denominator must be non-zero.
```

16-Create a Python program that determines the eligibility of a person for a senior citizen discount based on age using if-else.

In [72]:

```
age = int(input ("enter your age"))
price=int(input("enter price"))
if 60>= age >=40:
    discount = price*0.6

elif 40>=age >=30:
    discount = price*0.4

elif 30>=age>=18:
    discount=price*0.2

print("Age of senior citizen is :",age)
print("Discount of prise based on senior citizon age :",discount)
print("Net pay ",price-discount)
```

```
Age of senior citizen is : 40
Discount of prise based on senior citizon age : 3000.0
Net pay  2000.0
```

- we ask the senior citizon age .
- if senior citizon age is grater than or equal to 40 and less than 60 then discount based on his age is 60%.
- if senior citizon age is grater than or equal to 30 and less than 40 then discount based on his age is 40%
- if senior citizon age is grater than or equal to 18 and less than 30 then discount based on his age is 20%
- we use the print statements to print the net pay ,age and discount .

17-Write a Python program to categorize a given character as uppercase, lowercase, or neither using if-else.

In [73]:

```
character = input("enter the characters")

if character==character.upper():
    print(character,": is a upper case characters")

elif character==character.lower():
    print(character,": is a lower case characters")

else:
    print("character both in a lower case and upper case")
```

```
avneesh : is a lower case characters
```

lower case or upper case or in both,

- we use `if character==character.upper():` : this statements to check the character in a upper case or not.
- we use `elif character==character.lower():` :this statements to check the character in a lower case or not .
- we use the else statements to check the character in both lower case or not .

18-Write a Python program to determine the roots of a quadratic equation using if-else.

In [74]:

```
from math import sqrt

a=int(input("enter the numbers"))
b=int(input("enter the number of b"))
c=int(input("enter the number of c"))

# discriminant
disc=b**2 -4*a*c

if disc>0:
    x1=(-b+sqrt(disc))/(2*a)
    x2=(-b-sqrt(disc))/(2*a)
    print(x1)
    print(x2)
else:
    print("No solution")
```

```
0.0
-0.8
```

- We import the sqrt function from the math function .
- we ask the users input value of a,b,and c .
- we define the discriminant `(disc=b**2 -4*a*c)`
- we use the if statements if value of disc is grater than zero then print the x1 and x2.

19-Create a Python program that checks if a given year is a century year or not using if-else.

In [76]:

```
# Python3 code for find the century
# in a given year

def find_century(year):

    # No negative value is allow for year
    if (year <= 0):
        print("0 and negative is not allow for a year")

    # If year is between 1 to 100 it
    # will come in 1st century
    elif (year <= 100):
        print("1st century")
    elif (year % 100 == 0):
        print(year // 100,"century")
    else:
        print(year // 100 + 1,"century")
year = int(input("enter year"))
find_century(year)
```

```
21 century
```

- we define the function ,function name is find_century,that takes one

- `elif (year <= 100):` : this statements use to check the first century.
- `elif (year % 100 == 0):` : this statements to check the ony even years century.
- else statements use to check the odd year of centuary.

20-Write a Python program to determine if a given number is a perfect square using if-else.

In [111…

```
def is_perfect_square(n):
    i = 1
    n=int(input("enter the number"))
    while i * i <= n:
        if i * i == n:
            return n,":is a perfect square"
        i += 1
    return n ,":is not a perfect square"
is_perfect_square(n)
```

Out[111…

```
(25, ':is a perfect square')
```

21-Explain the purpose of the "continue" and "break" statements within if-else loops.

"continue" and "break" :-

- we use the continue function to skip the perticular value and after skip that value for loop and ,if elif and else condition work with continue, we use the break function to break the for loop ,while loop and if ,elif statements to perticular number.
- i'll explain this thing into the example.

In [78]:

```
for i in range(1,5):
    if i == 3:
        continue
    print(i)
```

```
1
2
4
```

In [79]:

```
for i in range(1,5):
    if i ==3:
        break
    print(i)
```

```
1
2
```

22-Create a Python program that calculates the BMI (Body Mass Index) of a person based on their weight and height using if-else.

In [81]:

```
weight = float(input("enter your weight in kg"))

height = float(input("enter your height in meters "))

BMI = weight/height**2
```

```
print("Your BMI is :",BMI)
```

```
Your BMI is : 22.22222222222222
```

- we ask the user weight in kg and height in meters.
- we use the `BMI = weight/height**2` :BMI formula to calculate the BMI of user .
- we use the print statements to print the BMI of user .

23-How can you use the "filter()" function with if-else statements to filter elements from a list?

In [85]:

```
score_list =[50,52,75,64,80,94,79,85,6,46]

def checkscore(n):
    if n>=80:
        return True
    return False
percentage_score=filter(checkscore,score_list)

score =list(percentage_score)
print("filterd score is:",score)
```

```
filterd score is: [80, 94, 85]
```

- we create the list of scores.
- we define the function ,function name is checkscore.
- if score is grater than the 80 return True if not grater than tha 80 return false.
- we use the filter function to filter the score grater then the 80 `percentage_score=filter(checkscore,score_list)` .
- we use the print statement to print the score.

24-Write a Python program to determine if a given number is prime or not using if-else.

In [118…

```
n=int(input("number"))

if n>1:
    for i in range(2,int(n/2)+1):
        if n%i==0:
            print(n,":is not a prime number")
        break
    else:
        print(n,": is a prime number")

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

```
1 : is not a prime number
```

- we ask the users enter the numbers that you want to check.
- we use the for loop to iterate the numbers from the n.
- inside for loop we use the multiple if ,else statements .

Map :-

1-Explain the purpose of the `map()` function in Python

- map is trying to do nothing but it is just trying to emplement a function map takes two arguments one is function and one is a data ,map is trying to maping element wise ,when ever i have to similar kind of function on the intire list or intire iterables then i can simply try to call the maper .

In [5]:

```
# example
lis=[1,2,3,4,5,6]
list(map(lambda a:a+2,lis))
```

Out[5]:

```
[3, 4, 5, 6, 7, 8]
```

2-Write a Python program that uses the `map()` function to square each element of a list of numbers.

In [54]:

```
lis=[1, 2, 3, 4, 5, 6]
list(map(lambda a:a**2,lis))
```

Out[54]:

```
[1, 4, 9, 16, 25, 36]
```

- first of all we create the list that takes sum intesers that we want to squere of them .
- we use the map function to actualy map function takes two arguments one is function and another is data here we use the lambda as a first arguments and list is a second arguments of map function
- `list` :we use this function as type casting of a map function.

3-How does the `map()` function differ from a list comprehension in Python, and when would you choose one over the other?

- List comprehension is more concise and easier to read as compared to map.
- List comprehension allows filtering. In map, we have no such facility. For example, to print all even numbers in range of 100, we can write [n for n in range(100) if n%2 == 0]. There is no alternate for it in map
- List comprehension are used when a list of results is required as map only returns a map object and does not return any list.
- List comprehension is faster than map when we need to evaluate expressions that are too long or complicated to express
- Map is faster in case of calling an already defined function (as no lambda is required).

4-Create a Python program that uses the `map()` function to convert a list of names to uppercase.

In [38]:

```
l=["Avneesh","kumar","pwskills"]

list(map(lambda a:a.upper(),l))
```

Out[38]:

```
['AVNEESH', 'KUMAR', 'PWSKILLS']
```

to convert into a upper case characters.

- we use the the map function to convert into a upper case .

5-Write a Python program that uses the `map()` function to calculate the length of each word in a list of strings.

In [55]:

```
l=["Avneesh","Verma","Pwskills"]
list(map(len,l))
```

Out[55]:

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

- first of all we create a list of string .
- we use the map function to find the length of the perticular string.

6- How can you use the `map()` function to apply a custom function to elements of multiple lists simultaneously in Python?

In [56]:

```
numbers=[1,2,3,4,56,6,7,8,9,5]
power=[1,2,2,3,4,5,6,7,8,9]
list(map(pow,numbers,power))
```

Out[56]:

```
[1, 4, 9, 64, 9834496, 7776, 117649, 2097152, 43046721, 1953125]
```

- we create the two lst that takes some integer values .
- we use the map function to power of numbrs

7-Create a Python program that uses `map()` to convert a list of temperatures from Celsius to Fahrenheit.

In [18]:

```
def celsius_to_fahrenheit(celsius):
    return (celsius * 9/5) + 32


    # Use map() to apply the conversion function to each temperature
    list(map(celsius_to_fahrenheit, celsius_temperatures))

    # Display the results
    print("Celsius Temperatures:", celsius_temperatures)
    print("Fahrenheit Temperatures:", fahrenheit_temperatures)
```

Out[18]:

```
[32.0, 50.0, 68.0, 86.0, 104.0]
```

In [17]:

```
list(map(celsius_to_fahrenheit,celsius_temperatures))
```

Out[17]:

```
[32.0, 50.0, 68.0, 86.0, 104.0]
```

8-Write a Python program that uses the `map()` function to round each element of a list of floating-point numbers to the nearest integer.

In [52]:

```
lst=[12.1,24.5,14.02,52.5,100.202]
list(map(round ,lst))
```

```
Out[52]: [12, 24, 14, 52, 100]
```

- we creat the list that store the some floting number,that we want to round off these numbers.
- we use the map function to find the round off of perticular numbers.

Reduce :-

1-What is the `reduce()` function in Python, and what module should you import to use it? Provide an example of its basic usage.

- Reduce is nothing but is a litil bit siilar to the map function like map function takes two arguments one if a function and one is iterable or data it is also take these two arguments but reduce function reducing the elements and collaps in a one result .Reduce function import from a `functools` library `(from functools import reduce )` :Reuce function is keep on reducing the values or number unless and untill task is not completed .Where ever i need to only one result in this situation reduce function is comes into a picture.

```
In [5]:
## Example :-

## import the reduce function
from functools import reduce
## create the list

l=[1,2,3,4,5,6]

## Use the reduce function.
reduce(lambda a,b: a*b,l)
```

```
Out[5]: 720
```

- first of all we import the reduce function from the functools library.
- we create the list of numbers .
- we use the reduce function to find the product of all elements in a list.

2-Write a Python program that uses the `reduce()` function to find the product of all elements in a list.

```
In [27]:
## Example :-

## import the reduce function
from functools import reduce
## create the list

l=[1,2,3,4,5,6]

## Use the reduce function.
reduce(lambda a,b: a*b,l)
```

```
Out[27]: 720
```

- first of all we import the reduce function from the functools library.

- we create the list of numbers .
- we use the reduce function to find the product of all elements in a list.

3-Create a Python program that uses `reduce()` to find the maximum element in a list of numbers.

In [30]:

```
l=[12,3,85,9,0,9,1,5]
## import the reduce function
from functools import reduce

## find the max of number

reduce(lambda a,b:max(a,b),l)
```

Out[30]:

```
85
```

- first of all we import the reduce function from the functools library.
- we create the list of numbers .
- we use the reduce function to find the maximum elements in a list of numbers.

4-How can you use the `reduce()` function to concatenate a list of strings into a single string?

In [31]:

```
## create the string list

l=["My","Name","Is","Avneesh","Verma"]

## import the function

from functools import reduce

reduce(lambda a,b:a+b,l)
```

Out[31]:

```
'MyNameIsAvneeshVerma'
```

- first of all we import the reduce function from the functools library.
- we create the list of strings that we want to concatenate .
- we use the reduce function to find the concatination of all strings in a list.

5-Write a Python program that calculates the factorial of a number using the `reduce()` function.

In [43]:

```
## we want to find the factorial of n
n=4
## use the reduse function to find the factorial of a number
fact=reduce(lambda a,b:a*b,range(1,n+1))

## call the fact variable
fact
```

Out[43]:

```
24
```

6-Create a Python program that uses `reduce()` to find the GCD (Greatest Common Divisor) of a list of numbers.

In [77]:

```
from functools import reduce
l=[12, 24, 27, 30, 36]

g=lambda a,b:a if b==0 else g(b,a%b)
print(reduce(lambda a,b:g(a,b),l))
```

```
3
```

- we import the reduce function from the functools library .
- create the list that contains the some integers.
- we use the reduce function to find the GCD of a list of numbers .

7-Write a Python program that uses the `reduce()` function to find the sum of the digits of a given number.

In [75]:

```
## create the list of numbers
l=[1,2,3,4,5,6]

#use the reduce function to find the value sum of numbers
reduce(lambda a,b:a+b,l)
```

Out[75]:

```
21
```

- first of all we import the reduce function from the functools library.
- we create the list of numbers .
- we use the reduce function to find the sum of all elements in a list.

Filter :-

1-Explain the purpose of the `filter()` function in Python and provide an example of how it can be used to filter elements from an iterable.

- filter function is nothing but its been used to filter some data,that filter function will do what it will filter out the data ased on some condition that we are going to aply this is where a filter function comes into a picture . filter is almost like a map function ,So filter is always to aply itself on every element of the iterables indivisualy where ever condition is going to be true it will keep only that data ,where ever condition is going to be false it will not keeping that perticular data in my final data.

In [47]:

```
## Example

s="Pw Skills"
list(filter(lambda a:a.islower(),s))
```

Out[47]:

```
['w', 'k', 'i', 'l', 'l', 's']
```

2-Write a Python program that uses the `filter()` function to select even numbers from a list of integers.

In [51]:

```
## create the list
```

```
l=[1,2,3,4,5,6,7,8,9,71,2,5]
## use the filter function
list(filter(lambda a:a%2==0,l))
```

```
Out[51]: [2, 4, 6, 8, 2]
```

- we create the list that stores some inteser value.
- we filter out the even numbers with the help of filter function.

3-Create a Python program that uses the `filter()` function to select names that start with a specific letter from a list of strings.

In [54]:

```
## create the list of string
l=["Pwskills","Sudh","Priya","krish"]
# i filter the string which starts with P

list(filter(lambda a:a[0]=="P",l))
```

```
Out[54]: ['Pwskills', 'Priya']
```

- first of aal we create the list of string .
- we use the filter function to filter out the data acourding to the specific latter .

4-Write a Python program that uses the `filter()` function to select prime numbers from a list of integers.

In [9]:

```
def is_prime(number):
    if number < 2:
        return False
    for i in range(2, int(number**0.5) + 1):
        if number % i == 0:
            return False
    return True


if __name__ == "__main__":
    # List of integers
    numbers = [2, 7, 15, 23, 10, 31, 50]

    # Use filter() to select prime numbers
    prime_numbers = list(filter(is_prime, numbers))

    # Display the results
    print("Original Numbers:", numbers)
    print("Prime Numbers:", prime_numbers)
```

```
Original Numbers: [2, 7, 15, 23, 10, 31, 50]
Prime Numbers: [2, 7, 23, 31]
```

- we create the function function function name is is_prime that take a agumenets as a list.
- we use the for loop to find the prime numbers.
- we use the print statements to print the prime numbers .

5-How can you use the `filter()` function to remove None values from a list in Python?

In [8]:

```
## create the list that takes none values

l=[1,None,2,None,3,None]

## use the filter function to remove the None values
list(filter(lambda a:a is not None,l))
```

Out[8]:

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

- We create the list of none values and inteser values .
- we use the filter function to filter out the inteser value and remove out the None values.

6-Create a Python program that uses `filter()` to select words longer than a certain length from a list of strings.

In [57]:

```
l=["Pwskills","Sudh","Priya","krish"]
list(filter(lambda a:len(a)>4,l))
```

Out[57]:

```
['Pwskills', 'Priya', 'krish']
```

- we create the list of the string .
- we use the filter function to filter the words longer than length 4 .

7-Write a Python program that uses the `filter()` function to select elements greater than a specified threshold from a list of values.

In [59]:

```
l=[1,2,3,4,5,26,2,6,26,2,6,26,2,5,154,51949,56,9,9]
list(filter(lambda a:a>=10,l))
```

Out[59]:

```
[26, 26, 26, 154, 51949, 56]
```

- we create the list that takes some inteser values .
- we use the filter function to filter out the data from the specific thresold.

Recursion :-

1-Explain the concept of recursion in Python. How does it differ from iteration?

- the basic meaning of recursion is nothing but calling a same function again and again that is the meaning of the recursion ,function is trying to call itself again and again unless and until task is not done.

- for loop iterate the values one by one but in this way function is not calling itself .loop will keep on going through all the elements one by one ,unless and until your task is not done.

In [101…

```
## ## example of recursion
def fact(n):
    if n==0:
        return 1
    elif n==1:
        return 1
    elif n<1:
        return "Factorial of this number is not possible"
    else:
        return n*fact(n-1)
fact(4)
```

Out[101…

```
24
```

2-Write a Python program to calculate the factorial of a number using recursion.

In [96]:

```
## create the function
def fact(n):
    if n==0:
        return 1
    elif n==1:
        return 1
    elif n<1:
        return "Factorial of this number is not possible"
    else:
        return n*fact(n-1)
```

In [97]:

```
fact(5)
```

Out[97]:

```
120
```

- first of all we create the function ,function name is fact and it is take one arguments that we wan to check the factorial of a number.
- if n is equal to the zero or 1 then return 1 beacuse factorial of zero and one is 1.
- we use the recursion to find the factorial of the number `n*fact(n-1)` : (recursion meanse function inside function).
- factorial of negative numbers is not possible.

3-Create a recursive Python function to find the nth Fibonacci number.

In [134…

```
def fib(n):
    if n==0:
        return n
    elif n==1:
        return 1
    elif n<0:
        return "Fibonaci of this number is not possible"
    else:
        return n+(n-1)+fib(n-2)
n=int(input("Plese enter the number"))
fib(n)
```

Out[134…

```
10
```

4-Write a recursive Python function to calculate the sum of all elements in a list.

In [38]:

```
def sum_arr(arr,size):
    if (size == 0):
        return 0
    else:
        return arr[size-1] + sum_arr(arr,size-1)
n=int(input("Enter the number of elements for list:"))
a=[]
for i in range(0,n):
    element=int(input("Enter element:"))
    a.append(element)
print("The list is:")
print(a)
print("Sum of items in list:")
b=sum_arr(a,n)
print(b)
```

```
The list is:
[1, 2]
Sum of items in list:
3
```

1. User must enter the number of elements in the list and store it in a variable.
2. User must enter the values to the same number of elements into the list.
3. The append function obtains each element from the user and adds the same to the end of the list as many times as the number of elements taken.
4. The list and the size of the list are passed as arguments to a recursive function.
5. If the size of the function reduces to 0, 0 is returned.
6. Otherwise the sum of the last element of the list along with the recursive function call (with the size reduced by 1) is returned.
7. The returned values are stored in a list and the sum of the elements in the list is printed.

5-How can you prevent a recursive function from running indefinitely, causing a stack overflow error?

- Ensure that your recursive function has a base case that will eventually be met, leading to the termination of the recursion. The base case defines when the function should stop calling itself. Without a proper base case, the recursion might continue indefinitely.

In [44]:

```
## example

def recursive_function(n):
    # Base case
    if n <= 0:
        return n

    # Recursive call
    recursive_function(n - 1)
```

6-Create a recursive Python function to find the greatest common divisor (GCD) of two numbers using the Euclidean algorithm.

In [45]:

```
def gcd_recursive(a, b):
    # Base case
    if b == 0:
        return a
    else:
        # Recursive case
        return gcd_recursive(b, a % b)

# Example usage:
num1 = 48
num2 = 18
```

```
result = gcd_recursive(num1, num2)

print(f"The GCD of {num1} and {num2} is: {result}")
```

```
The GCD of 48 and 18 is: 6
```

7-Write a recursive Python function to reverse a string.

In [23]:

```
def rev_str(s):
    if s=="":
        return ""
    else:
        return rev_str(s[1::])+s[0]
rev_str("Avneesh")
```

Out[23]:

```
'hseenvA'
```

- we create the function ,function name is rev_str that take one arguments as astring.
- if string is empty then reverse strring is also empty.
- we use the recursion to find the reverse of the string.

8-Create a recursive Python function to calculate the power of a number (x^n).

In [18]:

```
def power(a, b):
    if b != 0:
        return a * power(a, b - 1)
    else:
        return 1


a = 2
b = 3
print(a, "to the power", b, "is", power(a, b))
```

```
2 to the power 3 is 8
```

9-Write a recursive Python function to find all permutations of a given string.

In [21]:

```
def Permute(string):
    if len(string) == 0:
        return ['']
    prevList = Permute(string[1:len(string)])
    nextList = []
    for i in range(0,len(prevList)):
        for j in range(0,len(string)):
            newString = prevList[i][0:j]+string[0]+prevList[i][j:len(stri
            if newString not in nextList:
                nextList.append(newString)
    return nextList
Permute("ABC")
```

Out[21]:

```
['ABC', 'BAC', 'BCA', 'ACB', 'CAB', 'CBA']
```

10-Write a recursive Python function to check if a string is a palindrome.

In [15]:

```
def is_palindrom(s):
    if len(s)<1:
        return True
        return True
    if s[0]==s[-1]:
        return is_palindrom(s[1:-1])
    else:
        return False

s=str(input("Enter your string"))
if (is_palindrom(s)==True):
    print("string is palindrom")

else:
    print("string is not a palindrom")
```

```
string is palindrom
```

- we define the function ,function name is is_palindrom that takes string as asrgument that we have want to check string is a plindrom or not.
- we create the base condition is length of string is less than 1 in that situation string is always palidrom ,return True.
- if index of string zero is equal to index of -1 then it will be a palindom string , to check the whole string we use the recursive function (function inside the function).

11-Create a recursive Python function to generate all possible combinations of a list of elements.

In [7]:

```
def generate_combinations_recursive(nums):
    result = []
    if not nums:
        return [[]]

    for i in generate_combinations_recursive(nums[1:]):
        result.append(i)
        result.append([nums[0]] + i)
    return result
test_list = [55, 46, 47, 89, 76]
print(generate_combinations_recursive(test_list))
```

```
[[], [55], [46], [55, 46], [47], [55, 47], [46, 47], [55, 46, 47], [89], [5
5, 89], [46, 89], [55, 46, 89], [47, 89], [55, 47, 89], [46, 47, 89], [55,
46, 47, 89], [76], [55, 76], [46, 76], [55, 46, 76], [47, 76], [55, 47, 7
6], [46, 47, 76], [55, 46, 47, 76], [89, 76], [55, 89, 76], [46, 89, 76],
[55, 46, 89, 76], [47, 89, 76], [55, 47, 89, 76], [46, 47, 89, 76], [55, 4
6, 47, 89, 76]]
```

Basics of Functions :-

1-What is a function in Python, and why is it used?

- function is noting but it is a inbuilt logic or inbuilt sintexces which we try to use acieve the purpose that we have. we used function at the time of writing code to reusability of code , function is piece of code written to carry out specific task.To carry out that specific task, the function might or might not need multiple inputs. When the task is carried out, the function can or can not return one or more values.

2- How do you define a function in Python? Provide an example.

- We define the function with the help of def key word and provide the arguments if you want to .

In [124…

```
## example
 ## define function with the help of def keyword

def add_num(a,b):
    c=a+b
    print("sum of number a and b is :",c)
add_num(14,25)
```

```
sum of number a and b is : 39
```

3-Explain the difference between a function definition and a function call.

- functions are noting but it a kind of inbuilt logic or inbuilt sintexses which we can try to use achive a purpuse that we have.
- function call is an expresssion that passes controle and arguments (if any). function call use when execuite any function code .

4-Write a Python program that defines a function to calculate the sum of two numbers and then calls the function.

In [1]:

```
def sum_ofnumber(*n):
    result =0
    for i in n :
        result+=i
    print(result)
sum_ofnumber(1,2,3)
```

```
6
```

- we define the function ,function name is sum_ofnumbers ,it takes n number of arguments .
- we create the result variable ,inside this variable we store the result .
- we use the for loop to iterate the numbers through function .
- we increase the value of result by ading 1 .
- we call the function sum_ofnumber to print the sum of values .

5-What is a function signature, and what information does it typically include?

- In Python, a function signature provides crucial information about the types of parameters that a function can accept and the type of data it returns. The signature() function from the inspect module is used to determine the function signature, helping developers to ensure they are passing the correct types of arguments to their functions. This function can be particularly useful in large programs where a small mistake can lead to significant issues.

6-Create a Python function that takes two arguments and returns their product.

In [3]:

```
def add(a,b):
    c=a+b
```

```
add(10,20)
```

```
30
```

- we define the function,function name is add and it takes two arguments a and b that we want to product.
- we creat the variable c that hold the product of a and b .
- we use the print statements to print the value of c.

Function Parameters and Arguments :-

1-Explain the concepts of formal parameters and actual arguments in Python functions.

- `Formal arguments` : When a function is defined it (may) has (have) some parameters within the parentheses. These parameters, which receive the values sent from the function call, are called formal arguments.
- `Actual arguments` : The parameters which we use in the function call or the parameters which we use to send the values/data during the function call are called actual arguments

2-Write a Python program that defines a function with default argument values.

In [9]:

```
def defalt_function(a=10,b=20):
    c=a+b
    print(c)

defalt_function(a=100,b=25)
```

```
125
```

In [8]:

```
def defalt_function(a=10,b=20):
    c=a+b
    print(c)

defalt_function()
```

```
30
```

- we define the function ,function name is default_function that takes two default arguments a and b.
- we creat the c variable to store the some of result.
- one beautifull thing about the default value is if user not gives the input values then it takes default values or if user input the value then function takes these values .

3-How do you use keyword arguments in Python function calls? Provide an example.

In [13]:

```
def key_word(first_name,last_name):
    print("first name:",first_name)
    print("last name:",last_name)

key_word(first_name="pw",last_name="skills")
```

```
first name: pw
last name: skills
```

- we define the function ,function name is key_word it takes two key word arguments .
- here i'am taking name & arguments itself if you write it any order it will gives same result.

4-Create a Python function that accepts a variable number of arguments and calculates their sum.

In [17]:

```
def sum_of_num(*n):
    result=0
    for i in n:
        result+=i
    print(result)
sum_of_num(1,2,3,4,5,6,7,8,9,10)
```

```
55
```

- we define the function ,function name is sum_of_num it takes arguments .
- *n----- it means it take n number of arguments that we want to sum of them.
- we create the result variable to store the some of that numbers .
- we `result+=i` :use this function to add the looping value into the result variable.
- we call the function to some of `(1,2,3,4,5,6,7,8,9,10)` : these numbers .

5-What is the purpose of the `*args` and `**kwargs` syntax in function parameter lists?

`*args` :--------we use the *args in souch senarious where we do't know how many numbers of arguments put over here .

`**kwargs` :--------we use the ** in souch senarious where we want to take n numbers of key and value pairs.

Return Values and Scoping :-

1-Describe the role of the `return` statement in Python functions and provide examples.

- `return` ---- A return statements used to the execution of the function call and returns the result to the caller .The statements after the return statements are not execuite. return function basicly return the value it does't similar to print statements ,print statements basicly print the values print statements . In the industry label mostly use the return statements.

- variables are nothig is a place holder where we can store the some values or some logics .(like a=10) ,here a is variable that store the 10 value .
- `local variable` : If any variable create inside the function called local variable scope,local variable only execuite with the function call.(local variable not use any where in the program)
- `global variable` : if any variable create outside of the function that is called global variable global variable execuite any where in the program .

In [25]:

```
## `Exampl`:-

x="awesome"
def my_fun():
    x="funtastic"
    print("Local variable x:",x)
print("Global variable x :",x)
my_fun()
```

```
Global variable x : awesome
Local variable x: funtastic
```

3-Write a Python program that demonstrates the use of global variables within functions.

In [73]:

```
## Global variable
x=20
def add():

    ## local variable
    y=30
    ## sum of local and global variable.
    c=x+y
    print(c)
add()
```

```
50
```

4-Create a Python function that calculates the factorial of a number and returns it.

In [71]:

```
## factorial of a given number .
## method defining via the recursion

def fct(n):
    ## 1.se case condition
    if n<=1:
        return 1
    ## 2.cursivefunction call.
    else:
        return n*fct(n-1)

## driver code
n=5
result =fct(n)
print(result)
```

```
120
```

5-How can you access variables defined outside a function from within the function?

```
## Global variable
x=20
def add():

    ## local variable
    y=30
    ## sum of local and global variable.
    c=x+y
    print(c)
add()
```

```
50
```

Lambda Functions and Higher-Order Functions :-

1-What are lambda functions in Python, and when are they typically used?

- `Lambda Function` :-------Lambda is a noting but it is a inbuilt anonymous function (function without name),Arguments is a place holder . that is a variable that will be used to hold the value you want to pass into the function expression. A lambda function can have multiple variables depending on what you want to.
- lambda function is tipically used to where we need to write single line of code .
- where we need to save the space complexity in such senarious we use the lambda function.

2-Write a Python program that uses lambda functions to sort a list of tuples based on the second element.

In [113…

```
list_of_tuple=[("Hindi",30),("english",10),("math",60),("science",56)]

sort_list =sorted(list_of_tuple,key=lambda x:x[1])
```

In [114…

```
sort_list
```

Out[114…

```
[('english', 10), ('Hindi', 30), ('science', 56), ('math', 60)]
```

- we create the list of tuple that we want to sort based on the second element.
- we use the the lambda function to sort the list of tuple based on second element of the tuple.
- we use the sorted function sorted function take two arguments one is data and another is arguments based on this arguments sort the data.

3-Explain the concept of higher-order functions in Python, and provide an example.

- A function is called Higher Order Function if it contains other functions as a parameter or returns a function as an output i.e, the functions that operate with another function are known as Higher order Functions. It is worth knowing that this higher order function is applicable for functions and methods as well that takes functions as a parameter or returns a function as a

Properties of higher-order functions:

- A function is an instance of the Object type.
- You can store the function in a variable.
- You can pass the function as a parameter to another function.
- You can return the function from a function.
- You can store them in data structures such as hash tables, lists, ...

```
In [11]:
```