function in python

syntax of the function---

define the func()

print()

call func()

In [1]:
```
def greet(): # I define the function without Argument
    print('hello')# call the function without argument
    print('morning')
greet()
```

```
hello
morning
```

In [2]:
```
def greet(): # I define the function without Argument
    print('hello')# call the function without argument
    print('goodnight')
greet()
```

```
hello
goodnight
```

In [4]:
```
def greet():
    print('hello')
    print('good morning')
greet()
def great():
    print('hello')
    print('goodmorning')
greet()
```

```
hello
good morning
hello
good morning
```

always remember define the function top side

and call the fnction bottom side

In [7]:
```
def greet():
    print('hello')
    print('good morning')

def great():
    print('hello')
    print('goodmorning')

def greet():
    print('hello')
    print('good morning')

greet()
greet()
greet()
```

```
hello
good morning
hello
good morning
hello
good morning
```

when Declare the function without argument

and call the function without argument Remembwe

In [9]:
```
def greet():
    print('hello')
    print('good morning')
greet('milk')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[9], line 4
      2     print('hello')
      3     print('good morning')
----> 4 greet(      )

TypeError: greet() takes 0 positional arguments but 1 was given
```

In [11]:
```
def  add(x,y):
    c=x+y
    print(c)
add(3)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[11], line 4
      2     c=x+y
      3     print(c)
----> 4 add(3)

TypeError: add() missing 1 required positional argument: 'y'
```

In [12]:
```
def  add(x,y):
    c=x+y
    print(c)
add(3,4)
```

```
7
```

In [18]:
```
def add(x,y,z):# avoid print statement as much as
    c=x+y+z
    print(c)
add(2,3,4)
```

```
9
```

In [19]:
```
def add(x,y,z):
    c=x+y+z
    return c
add(2,3,4)
```

Out[19]: 9

In [26]:
```
def greet():
    print('hello')
    print('good morning')


def add(x,y,z):
    c=x+y+z
    return c

greet()
add(2,3,4)
```

```
hello
good morning
```

Out[26]: 9

In [29]:
```
def add(x,y):
    c=x+y
    return c
result=add(4,5)

print(result)
print(type(result))
```

```
9
<class 'int'>
```

In [36]:
```
def add_sub(x,y):
    c=x+y
    d=x*y
    return c,d
result=add_sub(4,5)

print(result)
print(type(result))
```

```
(9, 20)
<class 'tuple'>
```

In [37]:
```
def add_sub(x,y): #this function contain 2 operation
    c=x+y
    d=x*y
    return c,d
result,result1=add_sub(4,5)

print(result)
print(type(result))

print(result1)
print(type(result1))
```

```
9
<class 'int'>
20
<class 'int'>
```

In [39]:
```
def add_sub_mul(x,y): #this function contain 3 operation
    c=x+y
    d=x-y
    e=x*y
    return c,d,e
result,result1,result2=add_sub_mul(4,5)
print(result)
print(type(result))

print(result1)
print(type(result1))

print(result2)
print(type(result2))
```

```
9
<class 'int'>
-1
<class 'int'>
20
<class 'int'>
```

In [40]:
```
def fun():
    print('Welcome to GFG')
fun()
```

```
Welcome to GFG
```

In [43]:
```
def evenOdd(x):
    if(x%2 == 0):
        return 'Even'
    else:
        return'Odd'

print(evenOdd(16))
print(evenOdd(7))
```

```
Even
Odd
```

In [46]:
```
def myFun(x,y=50):
    print('x:',x)
    print('y:',y)
myFun(10)
```

```
x: 10
y: 50
```

In [48]:
```
def student(fname,lname):
    print(fname,lname)

student(fname='Greeks',lname='Pratice')
student(lname='Pratice',fname='Greeks')
```

```
Greeks Pratice
Greeks Pratice
```

In [49]:
```
def nameAge(name, age):
    print("Hi, I am", name)
    print("My age is ", age)

print("Case-1:")
nameAge("Suraj", 27)

print("\nCase-2:")
nameAge(27, "Suraj")
```

```
Case-1:
Hi, I am Suraj
My age is  27

Case-2:
Hi, I am 27
My age is  Suraj
```

In [50]:
```
def myFun(*args, **kwargs):
    print("Non-Keyword Arguments (*args):")
    for arg in args:
        print(arg)

    print("\nKeyword Arguments (**kwargs):")
    for key, value in kwargs.items():
        print(f"{key} == {value}")
```

```
# Function call with both types of arguments
myFun('Hey', 'Welcome', first='Geeks', mid='for', last='Geeks')
```

```
Non-Keyword Arguments (*args):
Hey
Welcome

Keyword Arguments (**kwargs):
first == Geeks
mid == for
last == Geeks
```

In [51]:
```
def f1():
    s='I love GreeksforGreeks'
    def f2():
        print(s)

    f2()
f1()
```

```
I love GreeksforGreeks
```

In [53]:
```
def f1():
    a='abhishek sahoo'
    def f2():
        print(a)
    f2()
f1()
```

```
abhishek sahoo
```

In [54]:
```
def square_value(num):
    """This function returns the square
    value of the entered number"""
    return num**2

print(square_value(2))
print(square_value(-4))
```

```
4
16
```

In [58]:
```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n* factorial(n-1)

print(factorial(4))
```

```
24
```

Formal Argument

Actual Argument--devided into 4-parts(position/keyword/Default/Variable Length)

In [59]:
```
def add(x,y): # x,y -formal argument
    c=x+y
    return c
add(5,6)# 5,6-Actual argument
```

Out[59]: 11

In [60]:
```
# positional argument

def person(name,age):
    print(name)
    print(age)
person('nit',21)
```

```
nit
21
```

In [61]:
```
def person(name,age):
    print(name)
    print(age)
person('nit')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[61], line 4
      2     print(name)
      3     print(age)
----> 4 person(     )

TypeError: person() missing 1 required positional argument: 'age'
```

In [62]:
```
def person(name):
    print(name)
    print(age)
person('nit',21)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[62], line 4
      2     print(name)
      3     print(age)
----> 4 person(     ,21)

TypeError: person() takes 1 positional argument but 2 were given
```

person give a 2 argument but user gaven 3 argument

In [64]:
```
def person(name,age):
    print(name)
    print(age)
person('nit',21,23)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[64], line 4
      2     print(name)
      3     print(age)
----> 4 person(     ,21,23)

TypeError: person() takes 2 positional arguments but 3 were given
```

person gave a 2 argument but user gave 0 argument

In [67]:
```
def person(name,age):
    print(name)
    print(age)
person()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[67], line 4
      2     print(name)
      3     print(age)
----> 4 person()

TypeError: person() missing 2 required positional arguments: 'name' and 'age'
```

In [68]:
```
def person(name,age): #missing argument
    print(name)
    print(age)
person(22)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[68], line 4
      2     print(name)
      3     print(age)
----> 4 person(22)

TypeError: person() missing 1 required positional argument: 'age'
```

In [69]:
```
def person(name,age): #missing argument
    print(name)
    print(age)
person(22,'nit')
```

```
22
nit
```

In [70]:
```
def person(name,age): #position considered
    print(name)
    print(age)
person(22,'nit')
```

```
22
nit
```

In [71]:
```
def person(name,age): #position considered
    print(name)# as  a string can not convert str to int
    print(age-1)
person(22,'nit')
```

```
22
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[71], line 4
      2     print(name)
      3     print(age-1)
----> 4 person(22,     )

Cell In[71], line 3, in person(name, age)
      1 def person(name,age): #position considered
      2     print(name)
----> 3     print(age-1)

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

In [73]:
```
x=str('abhi')
x
```

Out[73]: 'abhi'

positional argument--- always we need to match the position

keyward argument

In [87]:
```
def person(name,age):
    print(name)
    print(age+1)
person(age=22,name='nit')
```

```
nit
23
```

In [88]:
```
def person(name,age,phone):
    print(name)
    print(age+1)
    print(phone)
```

```
person(age=22,name='nit',phone=20303)
```

```
nit
23
20303
```

In [2]:
```
def person(name,age):
    print(name)
    print(age+1)


person(age=22,name='nit',0303)
```

```
  Cell In[2], line 6
    person(age=22,name='nit',0303)
                             ^
SyntaxError: leading zeros in decimal integer literals are not permitted; use an 0o
prefix for octal integers
```

bidefault argument

In [90]:
```
def person(name,age=18): #default argument
    print(name)
    print(age)
person('nit')
```

```
nit
18
```

In [91]:
```
def person(name,age=18): #default argument
    print(name)
    print(age)
person('nit',24)
```

```
nit
24
```

In [92]:
```
def person(name,age):
    print(name)
    print(age)
person('nit',24)
```

```
nit
24
```

In [93]:
```
def person(name,age=17):
    print(name)
    print(age)
person('nit')
```

```
nit
17
```

.Everytime we canot add only 2 value we can also pass more the 3 values .you can define the function when the number of argument are not fixerd

variable length argument

variable sayaing-----i declare 2 formal argument and 3 actual ardument my code should work

In [98]:
```
def sum(a,b): #i will apply(star) means-all
    c=a+b
    return c

sum(5,6,7,8)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[98], line 5
      2     c=a+b
      3     return c
----> 5 sum(5,6,7,8)

TypeError: sum() takes 2 positional arguments but 4 were given
```

In [102…
```
def sum(a, *b): #i will apply(star) means-all # we canot add operand
    #c=a+b
    print(a)
    print(type(a))
    print(b)
    print(type(b))

sum(5, 6,7,8,9,10)
```

```
5
<class 'int'>
(6, 7, 8, 9, 10)
<class 'tuple'>
```

In [105…
```
def sum(a,*b): #a=5/b=6,7,8/5=5+6=11+7=18+8=26
    c=a

    for i in b:
        c=c+i
    print(c)
sum(5,6,7,8)
```

```
26
```

In [106…
```
def sum(a,*b):
    c=0

    for i in b:
        c=c+i
```

```
    print(c)
sum(5,6,7,8)
```

```
21
```

In [107…

```
def sum(a,*b):
    c=1

    for i in b:
        c=c+i
    print(c)
sum(5,6,7,8)
```

```
22
```

In [ ]:

```
# server -collection of database
# cloud-data store in datacenter
# Database-(Container)stored the dsts and accesss the data
#DBMS-Application  modify the data and interactive the dsta
```