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

```
nit
22
```

In [2]:
```
def person(name,age):
    print(name)
    print(age)
person('nit')
```

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

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

In [3]:
```
def person(name,age):
    print(name)
    print(age)
person(22)
```

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

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

In [4]:
```
def person(name,age):
    print(name)
    print(age)
person('nit',22,23,45,56)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[4], line 4
      2     print(name)
      3     print(age)
----> 4 person('nit',22,23,45,56)

TypeError: person() takes 2 positional arguments but 5 were given
```

In [5]:
```
def person(name, age):
    print(name)
    print(age)

person(22,'nit')
```

```
22
nit
```

In [6]:

```python
def person(name, age):
    print(name)
    print(age-1)

person(22, 'nit')
```

```
22
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[6], line 5
      2     print(name)
      3     print(age-1)
----> 5 person(22, 'nit')

Cell In[6], line 3, in person(name, age)
      1 def person(name, age):
      2     print(name)
----> 3     print(age-1)

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

In [7]:

```python
def person(name, age):
    print(name)
    print(age-1)

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

```
nit
21
```

In [8]:

```python
def person(name, age, new_age ):
    print(name)
    print(age-1)

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

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[8], line 5
      2     print(name)
      3     print(age-1)
----> 5 person(age = 22, name = 'nit')

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

In [9]:

```python
def person(name, age, new_age ):
    print(name)
    print(age-1)
    print(new_age)

person(age = 22, name = 'nit', new_age=23)
```

```
nit
21
23
```

Default Argument

```
In [10]: def person(name,age=18):
             print(name)
             print(age)
         person('nit')
```

```
nit
18
```

```
In [11]: def person(name,age=18):
             print(name)
             print(age)
         person('nit',40)
```

```
nit
40
```

Variable Length Argument

```
In [12]: def person(name,age):
             print(name)
             print(age)
         person('nit',40,50,60,70,80)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[12], line 4
      2     print(name)
      3     print(age)
----> 4 person('nit',40,50,60,70,80)

TypeError: person() takes 2 positional arguments but 6 were given
```

```
In [13]: def sum(a,b):
             c = a+b
             print(c)
         sum(5,6,7,8)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[13], line 4
      2     c = a+b
      3     print(c)
----> 4 sum(5,6,7,8)

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

```
In [14]: def sum(a,*b):
             c = a+b
             print(c)
         sum(5,6,7,8)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[14], line 4
      2     c = a+b
      3     print(c)
----> 4 sum(5,6,7,8)

Cell In[14], line 2, in sum(a, *b)
      1 def sum(a,*b):
----> 2     c = a+b
      3     print(c)

TypeError: unsupported operand type(s) for +: 'int' and 'tuple'
```

In [16]:

```
def sum(a,*b):  #1st argument is fixed but for 2nd argument
    #c=a+b
    print(type(a))
    print(type(b))
sum(5,6,7,8)
```

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

In [17]:

```
def sum(a,*b): #1st argument is fixed & we fetch each value from the tuple & we can
    c=a
    for i in b:
        c=c+i
    print(c)
sum(5,6,7,8)
```

```
26
```

In [18]:

```
def sum(a,*b): #1st argument is fixed & we fetch each value from the tuple & we can
    c=a
    for i in b:
        c=c+i
    print(c)
sum(5,6,7,8,9,20)
```

```
55
```

In [19]:

```
def sum(a,*b): #1st argument is fixed & we fetch each value from the tuple & we can
    c=a
    for i in b:
        c=c+i
    print(c)
sum(5,6,7,8)
```

```
26
```

Kwargs

In [20]:

```
def person():
    person('ALEX', 36, 'JOHN', 987767)
```

In [21]:

```
def person(name, *data):
    print(name)
    print(data)

person('ALEX', 36, 'JOHN', 987767)
```

```
ALEX
(36, 'JOHN', 987767)
```

In [22]:

```
def person(name,*data):
    print('name')
    print(data)

person('ALEX', age = 36, home_place ='southcity', mob =987767)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[22], line 5
      2     print('name')
      3     print(data)
----> 5 person('ALEX', age = 36, home_place ='southcity', mob =987767)

TypeError: person() got an unexpected keyword argument 'age'
```

In [23]:

```
def person(name, **data):
    print('name')
    print(data)

person('ALEX', age = 36, home_place ='southcity', mob =987767)
```

```
name
{'age': 36, 'home_place': 'southcity', 'mob': 987767}
```

In [24]:

```
def person(name, **data):
    print('name')
    print(data)

person('ALEX', age = 36, home_place ='southcity', mob =987767, slary= 40000, marrie
```

```
name
{'age': 36, 'home_place': 'southcity', 'mob': 987767, 'slary': 40000, 'married': 'ye
s'}
```

Global Variable and Local Variable

In [25]:

```
a = 10  #-- globla variable

def something():
    b = 15 #local variable
```

In [26]:

```
a = 10  #-- globla variable

def something():
    b = 15 #local variable

    print('in function',b)
    print('out function',a)
```

In [27]:

```
a = 10
def something():
    b = 15
    print('in function',b)

print('out function',a)
```

```
out function 10
```

In [28]:

```
a = 10

def something():
    a = 15

print('in function',a)

print('out function',a)
```

```
in function 10
out function 10
```

In [29]:

```
a = 10

def something():
    b = 15
    print('in function',b)  # local variable
something()

print('out function',a) #gloabl variable
```

```
in function 15
out function 10
```

In [30]:

```
a = 10

def something():
    #if we remove this variable then can befault it consider as global variable
    print('in function',a)

something()
print('out function',a)
```

```
in function 10
out function 10
```

In [31]:

```
a = 10
def something():
    global a
    b = 15 # 15 is converted to local when user assigned global a
    print('in function',b)
    print('gloabl variable', a)

something()

print('out function',a)
```

```
in function 15
gloabl variable 10
out function 10
```

In [32]:

```python
a = 20

def something():
    global a
    a = 15
    print('in function',a)

    a = 15

something()
print('out function',a)
```

```
in function 15
out function 15
```

In [33]:

```python
x= 10  # Global variable

def update_x():
    global x  # Declare that we are using the global variable x
    x += 5    # Modify the global variable

update_x()
print(x)  # Output: 15
```

```
15
```

In [34]:

```python
x = 10  # Global variable

def update_x():
    globals()['x'] += 5  # Access and modify the global variable using the dictiona

update_x()
print(x)  # Output: 15
```

```
15
```

In [ ]:

```python

```