Functions

In [1]:
```
def am():
    print('Welcome Guyz')
```

In [2]:
```
def am():
    print('Welcome Guyz')
am()
```

```
Welcome Guyz
```

In [3]:
```
def greet():
    print('hello')
    print('my team')

```

In [4]:
```
def greet():
    print('hello')
    print('my team')
greet()
```

```
hello
my team
```

In [5]:
```
def greet():
    print('hello')
    print('my team')
greet()
def greet():
    print('hello')
    print('my team')
greet()
def greet():
    print('hello')
    print('my team')
greet()
```

```
hello
my team
hello
my team
hello
my team
```

In [6]:
```
def greet():

    print('hello myself khushant')

greet()
```

```
hello myself khushant
```

In [7]:
```
def greet():

    print('hello myself khushant')
greet()
greet()
greet()
```

```
hello myself khushant
hello myself khushant
hello myself khushant
```

In [8]:
```
def add(a,b):
    x = a+b
    print(x)

add(1,2,3,4)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[8], line 5
      2     x = a+b
      3     print(x)
----> 5 add(1,2,3,4)

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

In [9]:
```
def add(a,b):
    x = a + b
    print(x)

add(10,5)
```

```
15
```

In [10]:
```
def add(a,b,c):
    x = a + b + c + m
    print(x)

add(12,5,18)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[10], line 5
      2     x = a + b + c + m
      3     print(x)
----> 5 add(12,5,18)

Cell In[10], line 2, in add(a, b, c)
      1 def add(a,b,c):
----> 2     x = a + b + c + m
      3     print(x)

NameError: name 'm' is not defined
```

In [11]:
```
def add(a,b,c,m):
    x = a + b + c + m
    print(x)

add(12,5,18,4)
```

```
39
```

In [12]:
```
def greet():
    print('hello')
    print('my team')
greet()
```

```
hello
my team
```

In [13]:
```
def add(a,b):
    x = a + b
    print(x)

add(10,5)
```

```
15
```

In [14]:
```
def greet():
    print('hello')
    print('my team')
greet()

def add(a,b):
    x = a + b
    print(x)

add(10,5)
```

```
hello
my team
15
```

In [15]:
```
def greet():
    print('hello')
    print('my team')
def add(a,b):
    x = a + b
    print(x)

add(10,4)
greet()
```

```
14
hello
my team
```

In [16]:
```
def greet():
    print('hello')
    print('my team')

def add(a,b):
    x = a+b
    print(x)
```

```python
def sub(a,b):
    x = a-b
    print(x)

greet()
add(12,4)
sub(9,8)
```

```
hello
my team
16
1
```

In [17]:
```python
def add_sub(a,b):
    x = (a+b)
    y = (a-b)
    print(x)
    print(y)

add_sub(12,6)
```

```
18
6
```

In [18]:
```python
def add_sub(a,b):
    x = a+b
    y = a-b
    return x,y

add_sub(12,6)
```

Out[18]:
```
(18, 6)
```

In [19]:
```python
def add_sub(a,b):
    x = a+b
    y = a-b
    return x,y

result = add_sub(8,2)

print(result)
```

```
(10, 6)
```

In [20]:
```python
def add_sub(a,b):
    x = a+b
    y = a-b
    return x,y

result1,result2 = add_sub(9,7)
print(result1,result2)
```

```
16 2
```

Formal Argumentj & Actual Argument

```
In [21]: def boss(name,age):
             boss(name)
             boss(age)
         boss('khushant',21,35)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[21], line 4
      2     boss(name)
      3     boss(age)
----> 4 boss('khushant',21,35)

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

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

         boss('khushant',21)
```

```
khushant
21
```

```
In [23]: def boss(name,age):
             print(name)
             print(age+1)

         boss(21,'khushant')
```

```
21
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[23], line 5
      2     print(name)
      3     print(age+1)
----> 5 boss(21,'khushant')

Cell In[23], line 3, in boss(name, age)
      1 def boss(name,age):
      2     print(name)
----> 3     print(age+1)

TypeError: can only concatenate str (not "int") to str
```

Keyword Argument

```
In [24]: def boss(name,age):
             print(name)
             print(age+1)

         boss(age=21, name='khushant')
```

```
khushant
22
```

In [25]:
```python
def boss(name,age):
    print(name)
    print(age+1)

boss(age1=21, name='khushant')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[25], line 5
      2     print(name)
      3     print(age+1)
----> 5 boss(age1=21, name='khushant')

TypeError: boss() got an unexpected keyword argument 'age1'. Did you mean 'age'?
```

In [26]:
```python
def boss(name,age1):
    print(name)
    print(age1+1)

boss(age1=21, name='khushant')
```

```
khushant
22
```

In [27]:
```python
def boss(name,age,city):
    print(name)
    print(age)
    print(city)

boss(name='khushant', age=21, city='hyd')
```

```
khushant
21
hyd
```

In [28]:
```python
def boss(name, age=18):
    print(name)
    print(age)

boss('khushant', 21)
```

```
khushant
21
```

Variable Length Argument

In [1]:
```python
def sum(a,b):
    c = a+b
    return c

sum(5,8)
```

```
Out[1]:  13
```

```
In [2]:  def sum(a,b):
             c = a+b
             return c

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

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[2], line 5
      2     c = a+b
      3     return c
----> 5 sum(5,6,7,8,9,10)

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

```
In [3]:  def sum(a, *b):
             c = a+b
             return c

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

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[3], line 5
      2     c = a+b
      3     return c
----> 5 sum(5,6,7,8,9,10)

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

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

```
In [4]:  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 [5]:  def sum(a, *b): # 1st argument is fixed & we fetch each value from the tuple & we c
             c = a

             for i in b:
                 c = c + i
             print(c)

         sum(5,6,7,8,9,10,100,200,300)
```

```
645
```

In [6]:
```
def sum(a, *b): # 1st argument is fixed & we fetch each value from the tuple & we c
    c = a

    for i in b:
        c = c + i
    print(c)

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

```
26
```

- positional argument
- keyword argument
- default
- variable lenght (* at last arg) | (args)
- keyword + variable lenght(kwargs)

In [8]:
```
def person():
    person('ALEX', 36, 'JOHN', 987767)
```

In [9]:
```
def person(name, *data):
    print(name)
    print(data)

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

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

In [10]:
```
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[10], 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 [11]:
```
def person(name,**data):
    print('name')
    print(data)

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

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

Function Arguments we are completed

Global Variable vs Lvariableocal

In [12]:
```
a = 10

print(a)
```

```
10
```

In [13]:
```
a = 10

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

In [14]:
```
a = 10
def something():
    b = 15

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

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[14], line 5
      2 def something():
      3     b = 15
----> 5 print('in function',b)
      6 print('out function',a)

NameError: name 'b' is not defined
```

In [15]:
```
a = 10

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

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

```
out function 10
```

In [16]:
```
a = 10

def something():
    a = 15

print('in function',a)

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

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

In [17]:
```python
a = 10

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

something()

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

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

In [18]:
```python
a = 10  #globla var

def something():
    b = 55  # local var
    print('in function',b)
something()

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

```
in function 55
out function 10
```

In [19]:
```python
# if i want to define global variabel inside the function
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 [20]:
```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 [21]:
```python
x = 10  # Global variable

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

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

```
15
```

In [22]:
```
import keyword
keyword.kwlist
```

Out[22]:
```
['False',
 'None',
 'True',
 'and',
 'as',
 'assert',
 'async',
 'await',
 'break',
 'class',
 'continue',
 'def',
 'del',
 'elif',
 'else',
 'except',
 'finally',
 'for',
 'from',
 'global',
 'if',
 'import',
 'in',
 'is',
 'lambda',
 'nonlocal',
 'not',
 'or',
 'pass',
 'raise',
 'return',
 'try',
 'while',
 'with',
 'yield']
```

In [36]:
```
def count(lst):

    lst = [1,2,3,4,8,9,10]

lst
```

Out[36]:
```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]
```

In [31]:
```
def count(lst):
    even = 0
    odd = 0

    for i in lst:
        if i%2 == 0:
            even += 1
        else:
            odd +=1
    return even,odd

lst = [1,2,3,4,8,9,10]
even, odd = count(lst)

print(even)
print(odd)
```

```
4
3
```

In [32]:

```
def count(lst):

    even = 0
    odd = 0

    for i in lst:
        if i%2 == 0:
            even += 1
        else:
            odd +=1
    return even,odd

lst = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10,11,12,13]
even,odd = count(lst)

print("Even Number: {} and odd Number : {}".format(even,odd))
#format is function belongs to string & bydefault you need to pass any parameter
```

```
Even Number: 6 and odd Number : 7
```

In [33]:

```
# in progammin we need to continue these process thats why we need to use loop hear

def fib(n):
    a = 0
    b = 1

    print(a)
    print(b)

    for i in range(0, n):
        c = a + b
        a = b
        b = c

        print(c)

fib(10)
```

```
0
1
1
2
3
5
8
13
21
34
55
89
```

```
In [ ]:
```