In [52]:
```
def am9():
    print('Good Afternoon student')
```

In [53]:
```
def am9():
    print('Good Afternoon student')
am9()
```

```
Good Afternoon student
```

In [54]:
```
def greet():
    print('hello')
    print('good afternoon')
```

In [55]:
```
def greet():
    print('hello')
    print('good afternoon')
greet()
```

```
hello
good afternoon
```

In [56]:
```
def greet():
    print('hello')
    print('good afternoon')
greet()
def greet():
    print('hello')
    print('good afternoon')
greet()
def greet():
    print('hello')
    print('good afternoon')
greet()
```

```
hello
good afternoon
hello
good afternoon
hello
good afternoon
```

In [57]:
```
def greet():
    print('hello good morning boss')
greet()
```

```
hello good morning boss
```

In [58]:
```
def greet():
    print('hello good morning boss')
greet()
greet()
greet()
greet()
```

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

def add(x,y): c= x+y print(c) add(4,6,7,8)

In [60]:
```
def add(x,y):
    c = x+y
    print(c)

add(4,6)
```

```
10
```

def add(x,y,z): c=x+y+z+m print(c) add(1,4,5)

In [62]:
```
def add(x,y,z,m):
    c=x+y+z+m
    print(c)
add(1,4,5,7)
```

```
17
```

In [63]:
```
def greet():
    print('hello')
    print('good evening')
greet()
```

```
hello
good evening
```

In [64]:
```
def add(x,y):
    c=x+y
    print(c)
add(7,4)
```

```
11
```

In [65]:
```
def greet():
    print('hello')
    print('good morning')
greet()

def add(x,y):
    c = x+y
    print(c)
add(7,4)
```

```
hello
good morning
11
```

In [66]:
```
def greet():
    print('hello')
    print('good morning')
def add(x,y):
    c = x+y
    print(c)

add(7,4)
greet()
```

```
11
hello
good morning
```

In [67]:
```
def greet():
    print('hello')
    print('good evening')

def add(x,y):
    c=x+y
    print(c)

def sub(x,y):
    d=x-y
    print(d)

greet()
add(7,4)
sub(10,2)
```

```
hello
good evening
11
8
```

In [68]:
```
def add_sub(x,y):
    c= x+y
    d= x-y
    print(c)
    print(d)
add_sub(10,6)
```

```
16
4
```

In [69]:
```
def add_sub(x,y):
    c = x+y
    d = x-y
    return c,d

add_sub(10,6)
```

Out[69]: (16, 4)

In [70]:
```
def add_sub(x,y,e):
    c = x+y+e
    d = x-y-e
    return c,d,e

add_sub(10,7,4)
```

Out[70]: (21, -1, 4)

In [71]:
```
def add_sub(x,y):
    c = x+y
    d = x-y
    return c,d

add_sub(10,7)
```

Out[71]: (17, 3)

In [72]:
```
def add_sub(x,y):
    c=x+y
    d=x-y
    return c,d

result1,result2 = add_sub(7,5)

print(result1,result2)
```

```
12 2
```

In [73]:
```
def add(x,y):
    c = x+y
    print(c)
add(7,6)
```

```
13
```

Formal Argument & Actual Argument

def person(name,age): print(name) print(age) person('santosh',23,34)

In [74]:
```
def person(name,age):
    print(name)
    print(age)

person('santosh',23)
```

```
santosh
23
```

def person(name,age): print(name) print(age+1) person(23,'santosh')

Keyword

In [75]:
```
def person(name,age):
    print(name)
    print(age+1)

person(age=23, name='santosh')
```

```
santosh
24
```

def person(name,age): print(name) print(age+1) person(age1=23, name='santosh')

In [76]:
```
def person(name,age1):
    print(name)
    print(age1+1)

person(age1=23, name='santosh')
```

```
santosh
24
```

In [77]:
```
def person(name,age,city):
    print(name)
    print(age+1)
    print(city)

person(age=23, name='santosh', city = 'hyd')
```

```
santosh
24
hyd
```

In [78]:
```
def person(name,age=18):
    print(name)
    print(age)

person('santosh',24)
```

```
santosh
24
```

Variable Length Argument

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

sum(7,6)
```

Out[1]: 13

def sum(a, b): c = a+b return c sum(5,6,7,8,9,10) def sum(a, *b): c = a+b return c sum(5,6,7,8,9,10)

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 can add them.
    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 can add them.
    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 [7]:
```
def person():
    person('santosh', 23, 'sahu', 987767)
```

In [8]:
```
def person(name, *data):
    print(name)
    print(data)

person('santosh', 23, 'sahu', 987767)
```

```
santosh
(23, 'sahu', 987767)
```

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

person('santosh', age = 36, home_place ='chandrapur', mob =987767)

In [15]:
```
def person(name,**data):
    print('name')
    print(data)

person('santosh', age = 36, home_place ='chandrapur', mob =987767, edu = 'phd')
```

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

Functions arguments we are completed

global varaibe vs local variable

In [16]:
```
a = 10

print(a)
```

```
10
```

In [17]:
```
a = 10

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

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

In [19]:
```
a = 10

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

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

```
out function 10
```

In [20]:
```
a = 10

def something():
    a = 15

print('in function',a)

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

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

In [21]:
```
a = 10

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

something()

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

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

In [22]:
```
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 [23]:
```
# 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 [24]:
```
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 [25]:
```
x = 10  # Global variable

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

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

```
15
```

In [26]:
```
import keyword
keyword.kwlist
```

Out[26]:
```
['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 [38]:
```
def myfunc():
    lst = [1, 2, 3, 4, 8, 9, 10]
    print(lst)
myfunc()
```

```
[1, 2, 3, 4, 8, 9, 10]
```

In [35]:
```
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 [36]:
```
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 [37]:
```python
# 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 [ ]:
```python

```