In [2]:
```
def am9():
    print('Good Afternoon Student')
```

In [3]:
```
def am9():
    print('Good Afternoon Student')
am9()
```

```
Good Afternoon Student
```

In [4]:
```
def greet():
    print('hello')
    print('good afternoon')
```

In [5]:
```
def greet():
    print('hello')
    print('good afternoon')
greet()
```

```
hello
good afternoon
```

In [7]:
```
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 [8]:
```
def greet():
    print('hello good morning boss')
greet()
```

```
hello good morning boss
```

In [9]:
```
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
```

In [12]:
```
#def add(x,y): c=x+y print(c) add(4,6,7,8)
```

In [15]:
```
def add(x,y):
    c = x+y
    print(c)

add(1,4)
```

```
5
```

In [16]:
```
def add(x,y,z,m):
    c=x+y+z+m
    print(c)

add(1,4,5,7)
```

```
17
```

In [17]:
```
def greet():
    print('hello')
    print('good evening')

greet()
```

```
hello
good evening
```

In [18]:
```
def add(x,y):
    c = x+y
    print(c)
add(7,4)
```

```
11
```

In [19]:
```
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 [20]:
```
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 [22]:
```
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 [23]:
```
def add_sub(x,y):
    c= x+y
    d= x-y
    print(c)
    print(d)
add_sub(10,6)
```

```
16
4
```

In [24]:
```
def add_sub(x,y):
    c = x+y
    d = x-y
    return c,d
add_sub(10,6)
```

Out[24]: (16, 4)

In [25]:
```
def add_sub(x,y,e):
    c = x+y+e
    d = x-y-e
    return c,d,e

add_sub(10,7,4)
```

Out[25]: (21, -1, 4)

In [26]:
```
def add_sub(x,y):
    c = x+y
    d = x-y
    return c,d
add_sub(10,7)
```

Out[26]: (17, 3)

In [27]:
```
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 [28]:
```
def add(x,y):
    c = x+y
    print(c)
add(7,6)
```

```
13
```

Formal Argument & Actual Argument

In [ ]:
```
# def person(name,age): print(name) print(age) person('Adnan',23,34)
```

In [29]:
```
def person(name,age):
    print(name)
    print(age)
person('Adnan',23)
```

```
Adnan
23
```

In [ ]:
```
#def person(name,age): print(name) print(age+1) person(23,'Adnan')
```

Keyword

In [30]:
```
def person (name,age):
    print(name)
    print(age+1)

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

```
Adnan
24
```

In [ ]:
```
# def person(name,age): print(name) print(age+1) person(age1=23, name='Adnan')
```

In [31]:
```
def person (name ,age1):
    print(name)
    print(age1 + 1)

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

```
Adnan
24
```

In [32]:
```
def person(name,age,city):
    print(name)
```

```
    print(age+1)
    print(city)

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

```
santosh
24
hyd
```

In [33]:
```
def person(name,age=18):
    print(name)
    print(age)

person('santosh', 24)
```

```
santosh
24
```

Variable Length Argument

In [35]:
```
def sum(a, b):
    c = a+b
    return c

sum(7,6)
```

Out[35]: 13

In [36]:
```
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 [39]:
```
def sum(a, *b): # 1st argument is fixed & we fetch each value from the tuple & w
    c = a

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

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

```
11
18
26
35
45
145
345
645
```

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

    for i in b:
```

```
        c = c + i
    print(c)
sum(5,6,7,8)
```

```
26
```

- Positional argument
- keyword argumnet
- defualt
- variable lenght(* at last arg)| (args)
- keyword + variable length(kwargs)

In [3]:
```
def person():
person('Adnan', 23, 'Ansari', 987767)
```

In [4]:
```
def person(name, *data):
    print(name)
    print(data)

person('Adnan', 23, 'Ansari', 987767)
```

```
Adnan
(23, 'Ansari', 987767)
```

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

person('Adnan', age= 36, home_place = 'Bidar', mob=886747)

In [6]:
```
def person(name, **data):
    print('name')
    print(data)

person ('Adnan', age = 21 , home_place ='bidar' , mob = 886747, edu = 'phd')
```

```
name
{'age': 21, 'home_place': 'bidar', 'mob': 886747, 'edu': 'phd'}
```

Functions arguments we are completed

global variable vs local variable

In [7]:
```
a = 10

print(a)
```

```
10
```

In [8]:
```
a = 10

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

In [9]:
```
a = 10

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

In [10]:
```
a = 10

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

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

```
out function 10
```

In [11]:
```
a = 10

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

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

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

In [14]:
```
a = 10

def something():
    b = 15

    print('in function' , b)

something()

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

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

In [1]:
```
a = 10

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

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

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

In [2]:
```
a = 10

def something():
    global a
    b = 15 # 15 is conerted to loacal whe user assigned global a
```

```
    print ('in function', b)
    print ('global variable' , a)
something()
print('out function' , a)
```

```
in function 15
global variable 10
out function 10
```

In [3]:
```
x  = 10 # global variable
def update_x():
    global x
    x += 5
update_x()
print(x)
```

```
15
```

In [4]:
```
import keyword
keyword.kwlist
```

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

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

In [11]:
```
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 [14]:
```
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 strings & bydefault you need to pass any paramete
```

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

In [ ]:
```
def fib(n):
    a= 0
    b= 1

    print(a)
    print(b)
```

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

In [ ]: