Function Arguments

In [1]:
```
def greet(name, msg):
    """
    This function greets to person with the provided message
    """
    print(f"Hello {name} , {msg}")

#call the function with arguments
greet("Hemant", "Good Morning")
```

```
Hello Hemant , Good Morning
```

In [2]:
```
#suppose if we pass one argument

greet("iota") #will get an error
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-2-d22b5c3c0605> in <module>
      1 #suppose if we pass one argument
      2
----> 3 greet("iota") #will get an error

TypeError: greet() missing 1 required positional argument: 'msg'
```

Different Forms of Arguments

1. Default Arguments

We can provide a default value to an argument by using the assignment operator (=).

In [3]:
```
def greet( msg="Good Morning",name="Hemant"):
    """
    This function greets to person with the provided message
    if message is not provided, it defaults to "Good Morning"
    """
    print(f"Hello {name} , {msg}")

greet("Hemant")
```

```
Hello Hemant , Hemant
```

In [4]:
```
greet("Rajkumar","Good Night")
```

```
Hello Good Night , Rajkumar
```

In [5]:
```
#with out msg argument
greet("iota")
```

```
Hello Hemant , iota
```

Once we have a default argument, all the arguments to its right must also have default values.

def greet(msg="Good Morning", name)

will get a SyntaxError : non-default argument follows default argument

2. Keyword Arguments

kwargs allows you to pass keyworded variable length of arguments to a function. You should use **kwargs if you want to handle named arguments in a function

Example:

In [6]:
```
def greet(**kw):
    """
    This function greets to person with the provided message
    """
    #print(f"Hello {kwargs['name1']} , {kwargs['msg']}")
    print(kw)
    if kw:
        print(f"Hello {kw['name']} , {kw['msg']}, {kw['name1']}")
greet(name1 = "Hemant", msg="Good Morning", bye = "BYE",name = "Ram")
```

```
{'name1': 'Hemant', 'msg': 'Good Morning', 'bye': 'BYE', 'name': 'Ram'}
Hello Ram , Good Morning, Hemant
```

3. Arbitary Arguments

Sometimes, we do not know in advance the number of arguments that will be passed into a function.Python allows us to handle this kind of situation through function calls with arbitrary number of arguments.

Example:

In [44]:
```
def greet(msg = "Good Morning",*names):
    """
    This function greets all persons in the names tuple
    """
    print(names)

    for name in names:
        print(f"Hello,  {name}, {msg} ")

greet("satish", "murali", "naveen", "srikanth")
```

```
('murali', 'naveen', 'srikanth')
Hello,  murali, satish
Hello,  naveen, satish
Hello,  srikanth, satish
```

In [6]:
```
def fibo(n):
    an_2 = 1
    an_1= 1
    an=1
    for i in range(3,n+1):
        an = an_2+an_1
        an_2 = an_1
        an_1 = an
    return an

fibo(10)
```

Out[6]:
```
55
```

In [7]:
```
a10 = fibo(10)

print(f"a10 is {a10}")
```

```
a10 is 55
```