Types Of Argument

```
# Types of arguments in python(Over view Only)

# 1. Positional arguments
def power(a,b):
    return a**b
print(power(2,3))
# 2. Default arguments
def power(a = 1,b = 1):
    return a**b
print(power(3,4))
# 3. Keyword arguments
def power(a,b):
    return a**b
print(power(b = 5, a = 2))
# 4. Arbitary arguments
def power(*number):    # *number act as an tuple, which contain n number of parameters.
    temp = 1
    print(type(number))
    for i in number:
        temp *= i
    return temp
print(power(1,2,3,4))
```

1) Positional arguments

```
# 1. Positional Arguments
def display(name,age,address):
  print(name)
  print(age)
  print(address)



name = 'Hitendra'
age = 18
address = 'rajasthan'
display(name,age,address)
```

2) Keyword arguments

```
# 2. Keyword Arguments
def display(name,age,address):
  print(name)
  print(age)
  print(address)

name = 'Rakesh'
age = 24                                  #keyword arguments > normal assigned values
address = 'dehradun'
display(address = 'delhi',name = 'suresh',age = 30)  #position doesn't matter all actual arguments are complusory
```

Types Of Argument

3) Default arguments

```
#3. Default Arguments
def display(address,name = 'Hitendra',age = 18):
  print(name)                    #we can't set an default arguments in between of keyword arguments
  print(age)
  print(address)

name = 'hitendra'                #keyword arguments > default arguments
display(name = 'ramesh',address ="up")         #If we use default argumnets with keyword argumnets than
                                               #default argumnets value gets updated by keyword arguments
```

4) Arbitrary arguments

```
#4. Arbitary Arguments
def display(*name):                            #N numbers of actual arguments in single formal arguments
  for i in range(len(name)):                   #Accessing same as list with 0 indexing
    print(name[i])                        #advantges: doesn't remember the no. of actaul arguments

name = 'hitendra'
display(name,'rakesh','suresh','ramesh')   #both ways are correct direct and indirect values passing in actual arguments
#    name[0]     name[1]   name[2]   name[3]          #keyword direct assigning is not allowed within actual arguments for
                                                      #that condition we use keywords arbitary arguments(**)
```

5) Keyword Arbitrary arguments

```
#5. Keywords Arbitary Arguments
def display(**n):                    #pointers to the pointers for accessing actual argumnets
  print(n['name'])                   #N numbers of acutal/keyword arguemnts in single formal arguments
  print(n['age'])                    #Accessing name[default arguments names] for accessing an single values
  print(n)                           #advantges: doesn't remember the no. of actaul arguments
n = 'shanu'                          #return type is an dict
age = 18
address = 'rajasthan'                #direct passing is not allowed in actual arguemnts as we are using keywords
#indirect passing is allowed but later that value upadted by keyword arguments.(name = 'sahnu')
display(name = 'Himesh', age = 22, address = 'rawatbahta')
```

String Behavior

```
#string doesn't get updated while passing into an function as, strings are immutable
def fun1():
  a = "abc"+"d"


a = "abc"
print(a)
fun1()
print(a)
```

Types Of Argument

Tuples Behavior

```
#tuples doesn't get updated while passing into an function as, tuples are immutable
def fun1(tuple1):
  list1 = list(tuple1)
  list1.append(3)
  tuple1 = tuple(list1)
  print(tuple1)

tuple1 = (1,2)
print(tuple1)
fun1(tuple1)
print(tuple1)
```

Set Behavior

```
#set get updated while passing into fucntion as, sets are mutable.
def fun1(set1):
  set1.add(50)

set1 = {1,2,34,7,32,100,100}
print(set1)
fun1(set1)
print(set1)
```

List Behavior

```
#list get updated while passing into fucntion as, list are mutable.
def fun1(list1):
  list1.remove(1)

list1 = [1,2,3,4]
print(list1)
fun1(list1)
print(list1)
```

Dictionary Behavior

```
#Dictionary get updated while passing into fucntion as, Dictionary are mutable.
def fun1(dict1):
  dict1[101] = "hitendra"
  dict1[104] = "ujesh"

dict1 = {101:"rakesh",102:"ramesh",103:"suresh"}
print(dict1)
fun1(dict1)
print(dict1)
```