Python *args and **kwargs

In programming, we define a functionÂ to make a reusable code that performs similar operation. To perform that operation, we call a function with the specific value,Â this value is called a function argument in Python.

We would recommend you to read Python Function and Python Function Arguments.

Suppose, we define a function for addition of 3 numbers.

Example 1: Function to add 3 numbers

```
def adder(x,y,z):
    print("sum:",x+y+z)

adder(10,12,13)
```

When we run the above program, the output will be

```
sum: 35
```

In above program we have `adder()` function with three arguments *x*, *y* and *z*. When we pass three values while calling `adder()` function, we get sum of the 3 numbers as the output.

Lets see what happens when we pass more than 3 arguments in the `adder()` function.

```
def adder(x,y,z):
    print("sum:",x+y+z)

adder(5,10,15,20,25)
```

When we run the above program, the output will be

```
TypeError: adder() takes 3 positional arguments but 5 were given
```

In the above program, we passed 5 arguments to the `adder()` function instead of 3 arguments due to which we got `TypeError`.

Introduction to *args and **kwargs in Python

In Python, we can pass a variable number of arguments to a function using special symbols. There are two special symbols:

1. *args (Non Keyword Arguments)
2. **kwargs (Keyword Arguments)

We use **args* and ***kwargs* as an argument when we are unsure about the number of arguments to pass in the functions.

Python *args

As in the above example we are not sureÂ about the number of arguments that can be passed to a function. Python has **args* which allow us to pass the variable number of non keyword arguments to function.

In the function, we should use an asterisk `*` before the parameter name to pass variable length arguments.The arguments are passed as a tuple and these passed arguments make tuple inside the function with same name as the parameter excluding asterisk `*`.

Example 2: Using *args to pass the variable length arguments to the function

```
def adder(*num):
    sum = 0

    for n in num:
        sum = sum + n

    print("Sum:",sum)

adder(3,5)
adder(4,5,6,7)
adder(1,2,3,5,6)
```

When we run the above program, the output will be

```
Sum: 8
Sum: 22
Sum: 17
```

In the above program, we used **num* as a parameter which allows us to pass variable length argument list to the `adder()`Â function. Inside the function, we have a loop which adds the passed argument and prints the result. We passed 3 different tuples with variable length as an argument to the function.

Python **kwargs

Python passes variable length non keyword argument to function using **args* but we cannot use this to pass keyword argument. For this problem Python has got a solution called ***kwargs*, it allows us to pass the variable length of keyword arguments to the function.

In the function, we use the double asterisk `**` before the parameter name to denote this type of argument. The arguments are passed as a dictionary and these arguments make a dictionary inside function with name same as the parameter excluding double asterisk `**`.

Example 3: Using **kwargs to pass the variable keywordÂ arguments to theÂ functionÂ

```
def intro(**data):
    print("\nData type of argument:",type(data))

    for key, value in data.items():
        print("{} is {}".format(key,value))

intro(Firstname="Sita", Lastname="Sharma", Age=22, Phone=1234567890)
intro(Firstname="John", Lastname="Wood", Email="[emailÂ protected]", Country="Wakanda", Age=25, Phone=9876543210)
```

When we run the above program, the output will be

```
Data type of argument: <class 'dict'>
Firstname is Sita
Lastname is Sharma
Age is 22
Phone is 1234567890

Data type of argument: <class 'dict'>
Firstname is John
Lastname is Wood
Email is [emailÂ protected]
Country is Wakanda
Age is 25
Phone is 9876543210
```

In the above program, we have a function `intro()` with ***data* as a parameter. We passed two dictionaries with variable argument length to the `intro()` function. We have for loop inside `intro()` function which works on the data of passed dictionary and prints the value of the dictionary.

Things to Remember:

- **args* and ***kwargs* are special keyword which allows function to take variable length argument.
- **args* passes variable number of non-keyworded arguments and on which operation of the tuple can be performed.
- ***kwargs* passes variable number of keyword arguments dictionary to function on which operation of a dictionary can be performed.
- **args* and ***kwargs* make the function flexible.

Table of Contents