Python - Functions

Python - Functions

- A function is a block of organized, reusable code that is used to perform a single, related action.
- Function blocks begin with the keyword def followed by the function name and parentheses (( )).
- Any input parameters or arguments should be placed within these parentheses.
- The code block within every function starts with a colon (:) and is indented.
- The statement return [expression] exits a function, optionally passing back an expression to the caller. A return statement with no arguments is the same as return none.

- **Syntax**

```
def functionname( parameters ):
        "function_docstring"
        function_suite
        Return [expression]
```

Calling a Function

Defining a function only gives it a name, specifies the parameters that are to

```
# Function definition is here
def fun( str ):
   "This prints a passed string into this function"
   print str
  return;
# Now you can call printme function
fun("I'm first call to user defined function!")
fun("Again second call to the same function")
```

PASS BY REFERENCE VS VALUE

All parameters (arguments) in the Python language are passed by reference.

```
# Function definition is here
def changeme( mylist ):
   "This changes a passed list into this function"
   mylist.append([1,2,3,4]);
Print "Values inside the function: “ mylist
Return
# Now you can call changeme function
My list = [10, 20, 30];
Change me (mylist);
Print "Values outside the function: “mylist
```

Output:-

```
Values inside the function:  [10, 20, 30, [1, 2, 3, 4]]
Values outside the function:  [10, 20, 30, [1, 2, 3, 4]]
```

Function Arguments

You can call a function by using the following types of formal arguments −

1. Required arguments
2. Keyword arguments
3. Default arguments
4. Variable-length arguments

The *return* Statement

The statement return [expression] exits a function. A return statement with no arguments is the same as return None.

```
def sum( arg1, arg2 ):
   # Add both the parameters and return them."
   total = arg1 + arg2
   print "Inside the function : ", total
   return total;
```

Scope of Variables

All variables in a program may not be accessible at all locations in that program. This depends on where you have declared a variable.

Global variables

Local variables