FUNCTION

Creating & calling a Function(user defined)

A function is defined using the def keyword in python.

e.g.

```
def my_own_function():
   print("Hello from a function")

#program start here...
print("hello before calling a function")
my_own_function()                    #function calling
print("hello after calling a function")
```

Variable's Scope in function

There are two types of variables with the view of scope.

Local variable – accessible only inside the functional block where it is declared.	Global variable – variable which is accessible among whole program using global keyword.

Local variable program:
--- 	
def fun():
  s = "I love India!" #local variable
  print(s)
s = "I love World!"
fun()
print(s)
 	
Output:
I love India!
I love World!
	Global variable program:
	--- 	
	def fun():
	  global s #accessing/making global variable for fun()
	  print(s)
	  s = "I love India!“ #changing global variable's value
	  print(s)
	
	s = "I love world!"
	fun()
	print(s)
	
	Output:
	I love world!
	I love India!
	I love India!

Global variables in nested function

def fun1():
  x = 100
  def fun2():
  global x
  x = 200
  print("Before calling fun2: " + str(x))
  print("Calling fun2 now:")
  fun2()
  print("After calling fun2: " + str(x))
 	
fun1()
print("x in main: " + str(x))
	OUTPUT:
	
	Before calling fun2: 100
	Calling fun2 now:
	After calling fun2: 100
	x in main: 200

Parameters / Arguments

These are specified after the function name, inside the parentheses. Multiple parameters are separated by comma. The following example has a function with two parameters x and y. When the function is called, we pass two values, which is used inside the function to sum up the values and store in z and then return the result(z):

```
def sum(x,y):     #x, y are formal arguments  and parameters also
    z=x+y
    return z        #return the result

x,y=4,5
r=sum(x,y)        #x, y are actual arguments
print(r)
```

Note:-

A formal parameter, i.e. a parameter, is in the function definition.

An actual parameter, i.e. an argument, is in a function call.

Function Arguments----------Functions can be called using following types of formal arguments –

1. Positional arguments	**2. Default Argument**
Required arguments(Positional) - arguments passed to a function in correct positional order	Default arguments - that assumes a default value if a value is not provided to arguments
#Required arguments def square(x): z=x*x return z r=square() print(r) #In above function square() we have to definitely need to pass some value to argument x.	#Default arguments def sum(x=3,y=4): z=x+y return z r=sum() print(r) r=sum(x=4) print(r) r=sum(y=45) print(r) #default value of x and y is being used when it is not passed

Functions using libraries

Mathematical functions: Mathematical functions are available under math module. To use mathematical functions under this module, we have to import the module using import math. For e.g. import math r=math.sqrt(4) print(r) Functions available in Python Math Module: ceil(n), floor(n), log2(n),sin(n),sqrt(n),factorial(n),pow(n,y),tan(n),sin(n),fmod(x,y),exp(n)	**Functions using libraries(System defined function)** String functions: String functions are available in python standard module. capitalize(),count(),find(),index(),isalnum(),isupper(),islower(),isspace(),istitle(),isdigit(),isalpha(),replace(),title(),upper(),split(),swapcase(),splitlines() e.g. s="i love programming" r=s.capitalize() print(r)

Consider the following function calls with respect to the function definition. Identify which of these will cause an error and why?find the error(if any) in the following code and write the corrected code and underline it:

Def sum(a,b): returna+b print "the sum=" sum(7,-1)	def calculate(a,b=5,c=10): return a*b-c i) calculate(12,3) ii) calculate(c=50,35) iii) calculate(20, b=7, a=15) iv) calculate(x=10,b=12)
consider the following function headers. identify the correct statement:- (i)def correct(a=1,b=2,c): (ii) def correct(a=1,b,c=3): (iii) def correct(a=1,b=2,c=3): (iv) def correct(a=1,b,c):	complete the function body in the given code snippet: def f(number): #missing function body print(f(5))