```
Functions
==========
1.sum of two numbers
=================
def sum(x,y):
        "going to add two numbers"
        s=x+y
        print ("sum of two numbers is ",s)


>>> sum(40,50)
sum of two numbers is  90
>>>

2. another method
==================
>>> def fun2(x,y):
        print (" x + y",x +y)
        return x + y

>>> fun2(20,40)
 x + y 60
60
3. third method
=================
def sum(a,b):
            "Adding the two values"
         print "Printing within Function"
 print a+b
            return a+b
def msg():
            print "Hello"
            return

total=sum(10,20)
print ?Printing Outside: ?,total
msg()
print "Rest of code"

4. To print the fibanooci series

>>> def fib(n):    # write Fibonacci series up to n
...     """Print a Fibonacci series up to n."""
...     a, b = 0, 1
...     while a < n:
...         print(a, end=' ')
...         a, b = b, a+b
...     print()
...
>>> # Now call the function we just defined:
... fib(2000)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597

5. Simple function with parameter
==================================

def greet(name):

    """This function greets to
        the person passed in as
        parameter"""

       print("Hello, " + name + ". Good morning!")
```

```
Function call
===============
greet('Paul')

Doc string
=============
print(greet.__doc__)

6. Example program with return statement
=========================================

def absolute_value(num):

        """This function returns the absolute
        value of the entered number"""


        if num >= 0:

                return num

        else:

                return -num

#
 Output: 2

print(absolute_value(2))


# Output: 4

print(absolute_value(-4))


7. function with two arguments
================================

def greet(name,msg):

   """This function greets to
   the person with the provided message"""

   print("Hello",name + ', ' + msg)


greet("Monica","Good morning!")

8. Python function with default arguments
==========================================
def greet(name, msg = "Good morning!"):

   """
   This function greets to
   the person with the
   provided message.


   If message is not provided,
   it defaults to "Good
   morning!"
   """
```

```
   print("Hello",name + ', ' + msg)


greet("Kate")

greet("Bruce","How do you do?")

9. Write a function to print a string
====================================
# Function definition is here
def printme( str ):
   "This prints a passed string into this function"
   print str
   return;

# Now you can call printme function
printme("I'm first call to user defined function!")
printme("Again second call to the same function")

10. Program for pass by reference vs value
=========================================

# 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
mylist = [10,20,30];
changeme( mylist );
print "Values outside the function: ", mylist

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

11. Another example for pass by ref
==================================

def changeme( mylist ):
   "This changes a passed list into this function"
   mylist = [1,2,3,4]; # This would assig new reference in mylist
   print "Values inside the function: ", mylist
   return

# Now you can call changeme function
mylist = [10,20,30];
changeme( mylist );
print "Values outside the function: ", mylist

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

12. Required Arguments
======================

# Function definition is here
def printme( str ):
   "This prints a passed string into this function"
   print str
   return;
```

```
# Now you can call printme function
printme()

13. Keyword Arguments
=======================

# Function definition is here
def printme( str ):
   "This prints a passed string into this function"
   print str
   return;

# Now you can call printme function
printme( str = "My string")

Another example
================
# Function definition is here
def printinfo( name, age ):
   "This prints a passed info into this function"
   print "Name: ", name
   print "Age ", age
   return;

# Now you can call printinfo function
printinfo( age=50, name="miki" )

default arguments
==================

# Function definition is here
def printinfo( name, age = 35 ):
   "This prints a passed info into this function"
   print "Name: ", name
   print "Age ", age
   return;

# Now you can call printinfo function
printinfo( age=50, name="miki" )
printinfo( name="miki" )

variable length arguments
=========================

# Function definition is here
def printinfo( arg1, *vartuple ):
   "This prints a variable passed arguments"
   print "Output is: "
   print arg1
   for var in vartuple:
      print var
   return;

Arbitrary Arguments
====================

def greet(*names):
   """This function greets all
   the person in the names tuple."""

   # names is a tuple with arguments
   for name in names:
       print("Hello",name)

greet("Monica","Luke","Steve","John")
```

```
# Now you can call printinfo function
printinfo( 10 )
printinfo( 70, 60, 50 )

Anonymous Functions
=======================

# Function definition is here
sum = lambda arg1, arg2: arg1 + arg2;



# Now you can call sum as a function
print "Value of total : ", sum( 10, 20 )
print "Value of total : ", sum( 20, 20 )

Return Statement
================

# Function definition is here

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

# Now you can call sum function
total = sum( 10, 20 );
print "Outside the function : ", total

Global vs local variables scope
================================

total = 0; # This is global variable.

def sum( arg1, arg2 ):

   total = arg1 + arg2; # Here total is local variable.
   print "Inside the function local total : ", total
   return total;

# Now you can call sum function
sum( 10, 20 );
print "Outside the function global total : ", total

Recursive Functions
===================

# An example of a recursive function to
# find the factorial of a number

def calc_factorial(x):
    """This is a recursive function
    to find the factorial of an integer"""

    if x == 1:
        return 1
    else:
        return (x * calc_factorial(x-1))

num = 4
print("The factorial of", num, "is", calc_factorial(num))
```