Functions in Python

Basic Theory:

A function is a block of organized, reusable code that is used to perform a single, related action. Functions provide better modularity for your application and a high degree of code reusing.

Defining a Function

You can define functions to provide the required functionality. Here are simple rules to define a function in Python.

- 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. You can also define parameters inside these parentheses.
- The first statement of a function can be an optional statement - the documentation string of the function or docstring.
- 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]
```

By default, parameters have a positional behavior and you need to inform them in the same order that they were defined.

Example

The following function takes a string as input parameter and prints it on standard screen.

```
def printme( str ):
   "This prints a passed string into this function"
   print str
   return
```

Calling a Function

Defining a function only gives it a name, specifies the parameters that are to be included in the function and structures the blocks of code.

Once the basic structure of a function is finalized, you can execute it by calling it from another function or directly from the Python prompt. Following is the example to call printme() function –

```
# 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")
```

Pass by reference vs value

All parameters (arguments) in the Python language are passed by reference. It means if you change what a parameter refers to within a function, the change also reflects back in the calling function. For example-

```
# 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
```

Here, we are maintaining reference of the passed object and appending values in the same object. So, this would produce the following result –

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

There is one more example where argument is being passed by reference and the reference is being overwritten inside the called function.

```
# Function definition is here
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
```

The parameter mylist is local to the function changeme. Changing mylist within the function does not affect mylist. The function accomplishes nothing and finally this would produce the following result –

```
Values inside the function:  [1, 2, 3, 4]
```

```
Values outside the function:  [10, 20, 30]
```

Example:

1. Write a program in Python using function (recursive and non recursive) which will calculate factorial of a number using function. The number is being passed as argument.

```
x = input("Enter number:")
product = 1
def fact(x):
    pro = 1
    for i in range(1, x+1):
        pro *= i
    return pro
def factrec(x):
    product = 1
    if x == 1:
        return 1
    product = product * x * factrec(x - 1)
    return product
product = factrec(x)
print(product)
```

2. Write a program in Python which will calculate square root of a number using function (without using pow() and math.sqrt()). The number is being passed as argument.

```
x = input("Enter a number:")
def root(x):
    return (x ** 0.5)
print(root(x))
```

3. Write a program in Python which will calculate LCM of n numbers using function where n is given as input.

```
def find_lcm(num1, num2):
    if(num1>num2):
        num = num1
        den = num2
    else:
        num = num2
        den = num1
    rem = num % den
    while(rem != 0):
        num = den
        den = rem
        rem = num % den
    gcd = den
    lcm = int(int(num1 * num2)/int(gcd))
```

```
    return lcm

l = list(map(int, raw_input("Enter:").split()))
num1 = l[0]
num2 = l[1]
lcm = find_lcm(num1, num2)

for i in range(2, len(l)):
    lcm = find_lcm(lcm, l[i])

print(lcm)
```

4. Write a program in Python which will calculate factors of a number using function. The number is being passed as argument.

```
x = input("Enter number:")
def factors(x):
    for i in range(1, (x/2 + 1)):
        if x%i == 0:
            print(i)
factors(x)
```

Assignment:

1) Write a program in Python using function (recursive and non recursive) to generate Fibonacci series up to n^{th} term. The n is provided as input and passed to the function.
2) Write a program in Python using function to generate Pascal's triangle of n rows.
3) Write a program in Python which will calculate GCD of n numbers using function where n is given as input.
4) Write a menu driven Python program to perform basic mathematical operations. All the operations are defined as functions. The user can continue operation as long the user wants. The operations are addition, subtraction, multiplication, division, and exponentiation.
5) Write a Python program which calculates volume of a box using function. The number of arguments passed, are at most three and at least zero
6) Write a Python program using function which prints the name of the subjects you like to read. The total number of subjects may vary. The subject names are passed as arguments.
7) Write a Python program using function which calculates simple interest. The rate of interest for senior citizen is 12% and for others is 10%.
8) Write a Python program using function that computes P(n,r).
9) Write a Python program using function that computes C(n,r).
10) Write a Python program using function which finds out maximum and minimum of three numbers.
11) Write a Python program to count the number arguments passed as command line arguments and print the program name. Also print the arguments.