1. In Python, what is the difference between a built-in function and a user-defined function? Provide an example of each.

Build-in functions are predefined functions in python like print() and len().

User defined functions are the ones which we create using def keyword for code reusability and modularity.

Eg: def addition(a,b): #function to perform addition of two numbers

Return a+b

1. How can you pass arguments to a function in Python? Explain the difference between positional arguments and keyword arguments.

We can pass arguments to a function while calling the function. For example to call above addition function addition(3,4) can be passed.

Keyword arguments

Parameter Names are used to pass the argument during the function call. Eg: addition(a=3,b=4)

Positional arguments

Arguments are passed in the order of parameters. The order defined in the order function declaration. Eg: addition(3,4)

1. What is the purpose of the return statement in a function? Can a function have multiple return statements? Explain with an example.

Return statement ends the definition of a function and returns a value or expression to the caller.

Yes, a function can be defined with multiple return statements.

def addition(a,b):

return a+b

return a-b

return a\*b

addition(3,4)—7 will be returned as output only first return statement executes and value is sent to the caller.

1. What are lambda functions in Python? How are they different from regular functions? Provide an example where a lambda function can be useful.

A lambda function is a small anonymous function. A lambda function can take any number of arguments, but can only have one expression.

Lambda functions are anonymous so they cannot be referred anywhere else in the code and it cannot contain any statements and it returns a function object which can be assigned to any variable.

Regular functions on the other hand break our program into smaller and modular chunks. As our program grows larger and larger, functions make it more organised and manageable. They can be called and used anywhere we want.

Eg:

def myfunc(n):  
  return lambda a : a \* n  
  
mydoubler = myfunc(2)  
  
print(mydoubler(11))

1. How does the concept of "scope" apply to functions in Python? Explain the difference between local scope and global scope.

Scope defines if a certain variable can be accessed across the code or is limited to certain functions.

Global scope can be defined as when a variable defined outside a function, can be accessed by any code and within a function.

Local scope can be defined when a variable is defined inside a function and it’s scope is limited to the function and cannot be accessed outside the function.

1. How can you use the "return" statement in a Python function to return multiple values?

We can use list(), tuple() and function in return statement to return multiple values.

def fun1():

return list(range(0,10))

1. What is the difference between the "pass by value" and "pass by reference" concepts when it comes to function arguments in Python?

Pass by value:

In pass by value, the value of a function parameter is copied to another location of the memory. When accessing or modifying the variable within the function, it accesses only the copy. Thus, there is no effect on the original value.

Pass by reference

In pass by reference, the memory address is passed to that function. In other words, the function gets access to the actual variable.

1. Create a function that can intake integer or decimal value and do following operations:
   1. Logarithmic function (log x)
   2. Exponential function (exp(x))
   3. Power function with base 2 (2x)
   4. Square root

Code

import math

def operations(a):

l=math.log(a)

ex=math.exp(a)

p=math.pow(a,2)

s=math.sqrt(a)

return f"log(a)={l} exp(a)={ex} pow(a,2)={p} sqrt(a)={s} where a is {a}"

print(operations(4))

output

log(a)=1.3862943611198906 exp(a)=54.598150033144236 pow(a,2)=16.0 sqrt(a)=2.0 where a is 4

1. Create a function that takes a full name as an argument and returns first name and last name.

def display\_name(fullname):

return fullname.split()

display\_name(“Harry Potter”)

o/p- ['Harry', 'Potter']