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

**Built in function: A built in function will be part of an existing python library/package. A built in function can be used by importing the parent package.**

**Example: print(“This is a built in function”).**

**Whereas, a user defined function is a functionality that needs to be created from scratch in order to solve a use case that can not be solved with built in functions. A user-defined function can use built-in functions in its definitions.**

**Ex: def user\_defined\_function():**

**print(“This is a user defined function result”)**

**user\_defined\_function()**

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

**Arguments are passed during the function call in python.**

**Positional arguments values are passed in the same order in function call in which they are defined in function definition, however, in keyword arguments, the value of the arguments are passed along with the name of the keywords during the function call so order of arguments can change.**

**Ex: def add(x,y):**

**Print x+y**

**Positional call: add(2,3)**

**Key word call: add(x=2, y=3)**

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

**A return statement returns an output to the function call and terminates the function execution right after it.**

**No, a function can have only one return statement. This return can be of mixed data types if data structures are being returned.**

Ex: **def add(x,y):**

**c= x+y**

**return c**

**add(4,5)**

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

**Lambda functions are online anonymous/inline functions in python. They do not need ‘def’ keyword to be defined.**

Ex:

**Square = lambda x: x\*x**

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

**Scope of variables define if they can be accessed with a function or globally at any point in a program.**

**Global scope makes a variable accessible globally across the program while local**

**scope contains the variable existence limited within the function it has been defined**

**in.**

6. How can you use the "return" statement in a Python function to return multiple values? 7. What is the difference between the "pass by value" and "pass by reference" concepts when it comes to function arguments in Python?

**Using data structures in return statement, multiple values can be returned.**

**Pass by Value: When function is provided with immutable data type values like int, float etc. They can be manipulated within function scope without changing the value of out of function scope variable.**

**Pass by reference: Here, the mutable type data types (list, dic, set etc.) are passed to the function argument and any changes made to arguments within the function scope would change the original argument value as well.**

8. Create a function that can intake integer or decimal values and do following operations:

a. Logarithmic function (log x)

b. Exponential function (exp(x))

c. Power function with base 2 (2x)

d. Square root

**def math\_operations(x):**

**print(“Logrithmic”, math.log(x))**

**print(“Exponential”, math.exp(x))**

**print(“Power”, math.pow(2,x))**

**print(“square root”, math.sqrt(x))**

**math\_operations(2)**

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

**def name\_separator(x):**

**name=x.split()**

**first\_name=name[0]**

**last\_name=name[1]**

**return first\_name, last\_name**

**y=name\_separator("Rajesh Kumar")**

**print(y)**