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

   Ans:-

   - Built-in functions are pre-defined functions that are provided by python.

     Ex: len() function . It is used to determine length of sequence or collection.
     Code:

     ```
     my_list = [1, 2, 3, 4, 5]
     length = len(my_list)
     print(length)
     ```

     Output: 5

   - User-defined functions, are functions created by the user to perform specific tasks. They are created with the def keyword followed by chosen function name.

     Ex: Here's an example of a user-defined function named add_num() that adds two numbers and returns the result:
     Code:

     ```
     def add_num(a, b):
         return a + b
     result = add_num(5, 3)
     print(result)
     ```

     Output: 8

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

   Ans:-

   In Python, you can pass arguments to a function by using positional arguments and keyword arguments.

   Positional arguments must be included in the correct order. Keyword arguments are included with a keyword and equals sign.

   Examples:

   - ***Positional argument:***

     ```
     def stud(name, age):
         print(f"Hello {name}! You are {age} years old.")
     stud("raju", 25)
     ```

     When we call the function with stud("raju", 25), the "raju" is assigned to the name , and 25 is assigned to the age .

- ***Keyword argument:***

```
def stud(name, age):
    print(f"Hello {name}! You are {age} years old.")
stud(age=25, name="raju")
```

Here , we assign the arguments as age=25 and name="raju". The order doesn't matter because they are explicitly assigned based on their names.

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

Ans:-

A return statement ends the processing of the current function and returns control to the caller of the function.

Yes, a function can have multiple return statements. Once a return statement is executed, the function exits and no further code in the function is executed***.***

Ex:-

Code:

```
def get_grade(score):
if score >= 90:
    return "A"
elif score >= 80:
    return "B"
elif score >= 70:
    return "C"
elif score >= 60:
    return "D"
else:
    return "F"

result = get_grade(87)
print(result)
```

Output: B

As shown above, the get_grade() function uses multiple return statements to determine the letter grade for a given score. The function checks the value of the score with different ranges and returns the appropriate grade. If the score is 87, the function will encounter the second return statement (return "B") and immediately exit, returning the value "B" back to

the caller. The returned value will be stored in the result variable and printed, resulting in the output "B".

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

Ans:-

Lambda functions are also known as anonymous functions, they are small, one-line functions that don't have a name and are defined using the lambda keyword. They can take any number of arguments but can only have a single expression. The result of the expression is automatically returned.

Example:-

```
# Regular function
        def multiply(x, y):
            return x * y

        print(multiply(2, 3))
```

Output: 6

```
# Lambda function
        multiply_lambda = lambda x, y: x * y

        print(multiply_lambda(2, 3))
```

 # Output: 6

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

Ans:-

Scope refers to the region of the program where a variable is accessible.

When a variable is defined inside a function, it is said to have local scope, which means that it can only be accessed within that function.

On the other hand, a variable defined outside of any function has global scope, which means that it can be accessed from anywhere in the program.

Ex Local variable:-

```
def my_function():
    x = 20  # Local variable
    print(x)

my_function()
```

Output: 20

Ex Global Variable:-

```
global_var = 20  # Global variable

def my_function():
    local_var = 30  # Local variable
    print(local_var)
    print(global_var)  # Accessing global variable

my_function()  # Output: 30, 20

print(global_var)  # Output: 20

print(local_var)  # Error: NameError - 'local_var' is not defined .
```

This error is shown because we are trying to access the local variable outside the function.

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

Ans:- You can use the "return" statement to return multiple values by separating them with commas.

Ex:-

```
def multiple_values():
    # ...
    return value1, value2, value3

result = multiple_values()
print(result)
```

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

   Ans:-

   When you give function parameters via reference, you're just passing references to values that already exist.

   When you pass arguments by value, on the other hand, the arguments become independent copies of the original values.

8. ***Create a function that can intake integer or decimal value and do following operations:***
   a. ***Logarithmic function (log x)***
   b. ***Exponential function (exp(x))***
   c. ***Power function with base 2 (2^x)***
   d. ***Square root***

   Ans:-

```python
import math

def perform_operations(number):
    results = {}

    # Logarithmic function (log x)
    results['logarithm'] = math.log(number)

    # Exponential function (exp(x))
    results['exponential'] = math.exp(number)

    # Power function with base 2 (2^x)
    results['power'] = math.pow(2, number)

    # Square root
    results['square_root'] = math.sqrt(number)

    return results

input_value = float(input("Enter a number: "))
results = perform_operations(input_value)
print(results)
```

Output:-

```
Enter a number: 20
{'logarithm': 2.995732273553991, 'exponential': 485165195.4097903, 'power': 1048576.0, 'square_root': 4.47213595499958}
```

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

Ans:-

```
name = input("Your full name : ").split(" ")
if len(name) == 2:
    print("First Name: ", name[0].capitalize(), "\nLast Name: ",
name[1].capitalize(), sep = "")
else:
    print("ERROR: unknown number of names found!")
```

Output:-

```
Your full name : jetha gada
First Name: Jetha
Last Name: Gada
```