1. **Why are functions advantageous to have in your programs?**
   * Functions help in organizing code into reusable blocks.
   * They reduce redundancy and make the code easier to maintain.
   * Functions improve readability and allow for easier debugging.
   * They promote modularity, allowing different parts of the program to be developed and tested independently.
2. **When does the code in a function run: when it's specified or when it's called?**
   * The code in a function runs when the function is called, not when it's defined.
3. **What statement creates a function?**
   * The def statement is used to create a function in Python.
4. **What is the difference between a function and a function call?**
   * A **function** is a block of code that performs a specific task and may take parameters as input.
   * A **function call** is the code used to execute a function and can pass arguments to the function.
5. **How many global scopes are there in a Python program? How many local scopes?**
   * There is one global scope per program execution.
   * Local scopes are created whenever a function is called, and there can be multiple local scopes at a time, corresponding to each function call.
6. **What happens to variables in a local scope when the function call returns?**
   * Variables in a local scope are destroyed (or garbage collected) when the function call returns. They are not accessible outside of the function.
7. **What is the concept of a return value? Is it possible to have a return value in an expression?**
   * A **return value** is the value that a function call evaluates to. It is the output of the function's computation.
   * Yes, it is possible to have a return value in an expression. For example:

Python

result = add\_numbers(3, 5) # Here, the function call add\_numbers(3, 5) is an expression that evaluates to a return value.

1. **If a function does not have a return statement, what is the return value of a call to that function?**
   * If a function does not have a return statement, the function call evaluates to None.
2. **How do you make a function variable refer to the global variable?**
   * You can use the global keyword to declare that a variable inside the function is referring to the global variable with the same name.

Python

x = 10

def my\_func():

global x

x = 20

my\_func()

print(x) # Output: 20

1. **What is the data type of None?**
   * None is a data type of its own in Python. It represents the absence of a value.
2. **What does the sentence import areallyourpetsnamederic do?**
   * This module name does not exist. Trying to import it will raise a ModuleNotFoundError.
3. **If you had a bacon() function in a spam module, how would you call it after importing spam?**

python

import spam

spam.bacon()

1. **What can you do to save a program from crashing if it encounters an error?**
   * You can use error handling techniques like try and except blocks to catch and handle exceptions gracefully.
   * Logging errors can help in diagnosing and fixing issues without crashing the program.
2. **What is the purpose of the try clause? What is the purpose of the except clause?**
   * The **try** clause is used to enclose the code that might throw an exception.
   * The **except** clause is used to handle the exception that was raised in the try block.
   * Together, they form a **try-except** block that allows you to handle errors gracefully.