1. **Functions are advantageous to have in programs because they promote code reusability, modular design, and organization. They allow you to break down complex tasks into smaller, manageable parts, making your code easier to understand, maintain, and debug.**
2. **The code in a function runs when it is called by invoking its name in the program's execution flow. Functions are not executed when they are defined or specified, but rather when they are called or invoked during program execution.**
3. **The def statement is used to create a function in Python. It defines the function's name, parameters (if any), and the block of code that gets executed when the function is called.**
4. **A function is a block of reusable code that performs a specific task when called. It is defined using the def statement. On the other hand, a function call is the actual execution or invocation of a function by using its name followed by parentheses. It passes any required arguments and triggers the execution of the function's code.**
5. **In a Python program, there is one global scope, which is the outermost scope accessible throughout the entire program. Local scopes are created whenever a function is called, and they exist only within the function's block. So, the number of local scopes depends on the number of function calls made during the program's execution.**
6. **When a function call returns, the local variables defined within the function's scope are destroyed, and their values are no longer accessible. The memory occupied by these variables is released, and their existence is limited to the duration of the function call.**
7. **The concept of a return value refers to the value that a function can optionally send back or "return" to the caller. It provides a way for a function to convey a result or data back to the code that called it. Yes, it is possible to use a return value in an expression and assign it to a variable or use it directly in further computations.**
8. **If a function does not have a return statement, the return value of a call to that function is None. None is a special Python object representing the absence of a value. It is commonly used to indicate the absence of a meaningful return value from a function.**
9. **To make a function variable refer to the global variable with the same name, you can use the global keyword inside the function. By declaring a variable as global, any assignment to that variable within the function will affect the global variable with the same name.**
10. **The data type of None in Python is NoneType. It represents the absence of a value or a null value.**
11. **The sentence import areallyourpetsnamederic does not have any standard meaning in Python. It appears to be an arbitrary statement trying to import a module named "areallyourpetsnamederic," which is not a built-in module or a commonly used module.**
12. **If you had a bacon() function in a spam module, you can call it after importing spam**
13. **using the following syntax:**

**import spam**

**spam.bacon()**

**Here, spam is the module name, and bacon() is the function name within the spam module.**

1. **To save a program from crashing if it encounters an error, you can use error handling techniques like try and except blocks. By enclosing potentially error-prone code within a try block and providing appropriate except clauses, you can catch and handle exceptions gracefully, preventing the program from terminating abruptly.**
2. **The purpose of the try clause is to enclose code that may potentially raise an exception. It allows you to test a block of code for errors. The except clause**