1) **Why are functions advantageous to have in your programs?**

- Functions are advantageous because they allow you to **reuse code and make it more modular.**
- Functions promote **code organization and readability.**
- They enable you to **break down complex tasks into smaller, manageable pieces**.
- Functions can be tested and debugged separately, which **enhances code maintainability.**
- They help in **avoiding code duplication** and promote code reusability.

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 specified.
- Function definition specifies what code should be executed when the function is called.

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 reusable code that performs a specific task.
- A function call is the action of executing or invoking the function to perform its task.

5) **How many global scopes are there in a Python program? How many local scopes?**

- There is **only one global scope** in a Python program.
- Local scopes are created whenever a function is called, so the number **of local scopes depends on the number of function calls.**

6) **What happens to variables in a local scope when the function call returns?**

- When a function call returns, the local variables within that function are **destroyed.**
- The **memory occupied** by local variables is **freed**, and the variables cease to exist.

7) **What is the concept of a return value? Is it possible to have a return value in an expression?**

- The return value is the value that a function sends back to the caller.
- It is used to communicate the result or output of the function's execution.
- Yes, it is possible to have a return value in an expression. We can use the return value directly in expressions or assign it to a variable.

8) **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 return value of a call to that function will be **`None`.** `None` is a special object in Python that represents the absence of a value.

9) **How do you make a function variable refer to the global variable?**

- To make a function variable refer to the global variable, you can use the **`global` keyword before the variable name** inside the function.
- This allows the function to access and modify the global variable instead of creating a new local variable with the same name.

10) **What is the data type of None?**

- The data type of `None` is `NoneType`.
- It represents the absence of a value or the lack of a value.

11) **What does the sentence `import areallyourpetsnamederic` do?**

- The sentence `import areallyourpetsnamederic` imports a module named "areallyourpetsnamederic" in the Python program.

12) **If you had a `bacon()` feature in a `spam` module, what would you call it after importing `spam`?**

- After importing the `spam` module, you can call the `bacon()` function using the following syntax:

```
import spam
spam.bacon()
```

13) **What can you do to save a program from crashing if it encounters an error?**

- We can use exception handling to save a program from crashing when it encounters an error.
- By placing code that may raise an exception inside a `try` block and providing appropriate handling using the `except` block, one can gracefully handle errors and prevent the program from abruptly terminating.

14) **What is the purpose of the try clause? What is the purpose of the except clause?**

- The purpose of the `try` clause is to enclose the code that may raise an exception.
- It allows you to test a block of code for errors.
- The purpose of the `except` clause is to specify the code that should be executed if a particular exception is raised in the `try` block.

- It catches the exception and handles it, preventing the program from crashing.