1. Why are functions advantageous to have in your programs?

2. When does the code in a function run: when it's specified or when it's called?

3. What statement creates a function?

4. What is the difference between a function and a function call?

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

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

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

8. If a function does not have a return statement, what is the return value of a call to that function?

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

10. What is the data type of None?

11. What does the sentence import areallyourpetsnamederic do?

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

13. What can you do to save a programme from crashing if it encounters an error?

14. What is the purpose of the try clause? What is the purpose of the except clause?

Ans:  
1. Functions are advantageous in programs for several reasons:

- \*\*Modularity\*\*: They allow you to break down your code into smaller, reusable chunks, making it easier to manage and understand.

- \*\*Reuse\*\*: Functions can be called multiple times from different parts of the program, reducing code duplication.

- \*\*Abstraction\*\*: They help in abstracting away implementation details, allowing you to focus on the high-level logic of your program.

- \*\*Organization\*\*: Functions make your code more organized and easier to maintain.

2. The code in a function runs when it is called, not when it is specified.

3. The `def` statement is used to create a function in Python.

4. A function is a block of reusable code that performs a specific task, whereas a function call is the act of invoking that function to execute its code.

5. In a Python program, there is one global scope and potentially multiple local scopes. Each function defines its own local scope.

6. Variables in a local scope cease to exist (are destroyed) when the function call returns. They are only accessible within the function's body.

7. A return value is the value that a function returns after its execution. It represents the output of the function's computation. Yes, it is possible to have a return value in an expression.

8. If a function does not have a return statement, the return value of a call to that function is `None`.

9. To make a function variable refer to a global variable, you can use the `global` keyword inside the function.

10. The data type of `None` is `NoneType`.

11. The statement `import areallyourpetsnamederic` attempts to import a module named `areallyourpetsnamederic` into the Python script. If such a module exists, its contents can be accessed within the script.

12. If you had a `bacon()` feature in a `spam` module, after importing `spam`, you would call it using `spam.bacon()`.

13. To save a program from crashing if it encounters an error, you can use exception handling. This involves wrapping the code that might raise an error within a `try` block and handling the error gracefully using an `except` block.

14. The purpose of the `try` clause is to enclose code that might raise an exception. If an exception occurs within the `try` block, the program execution moves to the corresponding `except` block. The `except` clause is used to handle the exception by providing appropriate instructions or actions to take when an exception occurs.