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

-> ***Functions*** are reusable pieces of code that can be called using a function’s name. Its advantages are as follows: -

* Functions allow the same piece of code to run multiple times
* Functions break long programs up into smaller components
* Functions can be shared and used by other programmers

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 it is called.***

3. What statement creates a function?

-> A function is created using the keyword ***def .***

For example: def my\_function():

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

-> A ***function*** is a piece of code that can be run multiple times by calling the function name.

A ***function call*** means asking the program to execute that function. A function can be called multiple times as required.

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

-> The scope associated with the ***\_main\_ module*** is termed as a global scope. For any python program, there can be only ***one global scope***.

The ***local scope*** in a python program is defined for a block of code such as ***function***. Each function in a python program has its own local scope in which all its variables and object names are defined. There can be ***multiple*** local scopes in a Python program.

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

-> The local scope of a function is loaded when the function is called by any other function. Once the function terminates, the local scope associated with it is also terminated. Hence the ***variables*** are also ***terminated***.

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

-> The ***return*** keyword in Python exits a function and tells Python to run the rest of the main program. A return keyword can send a value back to the main program.

While values may have been defined in a function, you can send them back to your main program and read them throughout your code.

***Yes***, return value can be used in an expression.

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, then the return value is ***None***.

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

-> To refer to the global variable from within a function, use the ***global()*** keyword.

10. What is the data type of None?

-> Data type of None is ***NoneType***.

11. What does the sentence import areallyourpetsnamederic do?

-> ***import*** keyword is used to import modules. Here it imports a module named areallyourpetsnamederic

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

-> import spam

***spam.bacon()***

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

-> ***Try and Except*** statement is used to handle errors in our Python code.

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

-> The ***try*** block is the block of statements you'd like to try executing. However, there may be runtime errors due to an exception, and this block may fail to work as intended.

The ***except*** block is triggered when the try block fails due to an exception. It contains a set of statements that often give you some context on what went wrong inside the try block.