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

**Ans:- Functions let us reuse the code many times by simply calling them as many times as we want. It’s also easy to remember what a specific piece of code does by creating a function with relatable name.**

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

**Ans:- A function only runs when we call it.**

1. What statement creates a function?

**Ans:- def *function\_name* (*parameters*):**

**The above syntax is used to create a function followed by its body.**

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

**Ans:- function defines the following:**

* **Code block that will be executed upon a call**
* **Values that function takes as argument during call**
* **Value that function returns back to the calling function**

**In simple terms, a function has a specific objective which it fulfils upon calling.**

**A function call ensures that the function body is executed. If the function definition has parameters in it, then while calling the function we are supposed to pass argument values(keeping the data type and number of arguments in mind).**

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

**Ans:- There is exactly 1 global and 1 local scope in Python. Any variable defined in a function or code block, that can only be accessed locally within same block/function is a local variable. A local variable defined in a function can be access inside the inner function too. However, scope of a local variable can be changed to global by prefixing ‘global’ to variable name. Ex:**

**def dummyFunction():**

**global a**

**a=10**

**return a**

**dummyFunction()**

**print(a)**

**In the above code, by default the variable ‘a’ was defined as local but after prefixing ‘global’ to it, the scope had changed to global. That’s the reason how we’re able to get 10 as output.**

**Global variables are generally defined inside the code without any enclosing function/block and can be accessed from anywhere.**

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

**Ans:- A variable defined in local scope is not accessible from outside of that function. So once the function returns, that variable will no longer be accessible. Any attempt to access it outside of its scope would lead to ‘NameError’.**

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

**Ans:- A function can either perform certain tasks without returning any value at all but in situations where we need processed information we can let the function return some value.**

**Example –**

**def add(a,b):**

**print(a+b)**

**the above function would accept 2 values and print their sum.**

**If we want to use the sum in further calculation, we need to let the add() function return its sum.**

**def add(a,b):**

**return a+b**

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

**Ans:- None**

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

**Ans:- A global variable has the widest accessibility so accessing it inside a function is possible. The below code gives an example:**

**aa=101**

**def xyz():**

**aaa = aa**

**xyz()**

**print(aa)**

**Here, ‘aa’ would refer to ‘aaa’.**

10. What is the data type of None?

**Ans:- NoneType**

11. What does the sentence import areallyourpetsnamederic do?

**Ans:- The above statement imports the code that exists in ‘areallyourpetsnamederic’ module. Using this statement, we will be able to utilise the functions/variables in this module.**

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

**Ans:-**

**import spam**

**spam.bacon()**

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

**Ans:- If we are well aware of the possible errors in our program, we can use exception handling to avoid crashing program.**

**We keep the error prone code in try block and define an except block too. Whenever an error occurs in try block, the control is sent to except block and statements in except block are executed. In such situation, the direction of execution changes without having the program crashed.**

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

**Ans:- Try block contains all the statements that may throw an error. In general, the program crashes whenever it comes across an error. In order to avoid crashing of program, the error prone statements are kept in try block and the code that we want to be executed in occurrence of error is enclosed in except block. Once the try block encounters an error, a corresponding except block gets executed. There could be multiple except blocks for one try block. Errors like division by zero, index out of bound:- can be handled by exception handling in python.**