**Note:**  Answers are given in blue

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

Functions help in **code re-usabilit**y and **code modularity** , which means that the entire code is organized into separate blocks, each of which is self-contained and performs a different task. This makes each block implementation and debugging much easier. The function can be reused countless times after it is defined. Using a function, **it is possible to reduce the size of a program** by calling and using the function at different places in the program.

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

The function runs when it is called.

1. What statement creates a function?

In Python, you define a function with the **def** keyword, then write the function identifier (name) followed by parentheses and a colon. The next thing you have to do is make sure you indent with a tab or 4 spaces, and then specify what you want the function to do for you

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

A function is a block of code that performs a single, related action.

A function call is an expression containing the function name followed by the function call operator, (). If the function has been defined to receive parameters, the values that are to be sent into the function are listed inside the parentheses of the function call operator. The argument list can contain any number of expressions separated by commas. It can also be empty.

Thus when the python interpreter comes across ‘a call’ to a predefined function, it transfers control to the function, and once the the block of code in that specific function are executed and when its when its return statement is executed it transfers control back to the main program.

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

There is one global scope in Python a program. The global scope can be accessed throughout the program and contains variables, functions and classes defined at the top level of the program. The global scope is the outermost scope of the program.

Local scopes are created whenever a function is called or when entering a block of code such as a loop or conditional statement. Each function or block has its own local scope, which is independent of other scopes. Local scopes are used to store variables specific to that particular function or block.

So, the number of global scopes in a Python program is one, while the number of local scopes can vary depending on the number of function calls and blocks of code that create local scopes.

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

When a function call returns in Python, the local scope associated with that function is destroyed, and the variables defined within that scope cease to exist; however, any changes made to mutable objects (e.g., lists, dictionaries) within the local scope will persist beyond the scope's lifetime. This is because the objects themselves exist in the global or outer scope, and only their references are passed to the local scope, but the local variables themselves will be inaccessible after the function call returns.

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

The return value refers to the value/result that the function provides after it has been called and executed and returns the value back to the caller. It is optional, i.e. depending on the use case, the programmer can choose whether a function returns a value or not. The return value can be of any data type in Python. A return value can be used in an expression. This means one can directly use the return value of a function call as part of a larger expression or assign it to a variable.

1. 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, or if the return statement is not executed during the function's execution, the function will return **None**.

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

To make a function variable refer to a global variable in Python, you can use the global keyword within the function. The global keyword allows you to declare that a variable within the function should refer to the global variable with the same name.

**Eg:**

person = ‘Clark Kent’

def superhero ():

global person

person = ‘Superman’

print (person) *#output: Clark Kent*

superhero ()

print (person)  *#output: Superman*

1. What is the data type of None?

The data type of None in Python is NoneType. NoneType is a built-in type in Python and represents the absence of a value or the lack of any specific type.

1. What does the sentence import areallyourpetsnamederic do?

The sentence import areallyourpetsnamederic tells the Python interpreter to bring in the module/package areallyourpetsnamederic.

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

Consider the following example:

import spam #we start by ‘importing’ the spam module

some\_variable = spam.bacon() #we call the bacon feature contained in the spam module and assign

# it to somevariable

print(some\_variable)

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

To prevent a program from crashing when encountering an error, you can implement error handling using try-except blocks. By wrapping potentially problematic code within a try block and specifying exception handling in an except block, you can handle errors gracefully and continue program execution.

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

The purpose of the try clause in a try-except block is to enclose a block of code where an exception or error might occur. The code inside the try block is executed, and if an exception is raised during its execution, the flow of control is transferred to the appropriate except block.

The purpose of the except clause is to define the actions to be taken if a specific exception is encountered within the try block. When an exception occurs, the program looks for an except block that matches the raised exception's type. If a match is found, the code inside the corresponding except block is executed. The except clause allows you to handle exceptions by providing error-specific handling logic.

Together, the try and except clauses provide a mechanism for handling exceptions and preventing program crashes by elegantly handling errors. Thus, enhancing the overall reliability and robustness of the program.