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

Ans: Advantages of functions:

1. Reusability
2. Scalability
3. Abstraction
4. Modularity
5. Encapsulation

Are some of the advantages of having functions in your programs.

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

Ans: A code is defined when it is specified, but it is run when it is called.

3. What statement creates a function?

Ans:

Example:

This statement creates the block of the function to be defined “def my\_function1():”

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

Ans:

A function is a self-contained block of code that performs a specific set of tasks. Functions are defined with a name, a set of parameters(if needed), and a body of code that defines what the function does when it is called.

A function call is the act of using a function by its name and providing any necessary arguments or defined within that function.

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

Ans:

There can be only one global scope in a python program, i.e. the top-level scope in a python program.

Multiple local scopes are created whenever you define a function or method.

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

Ans: When a function call in Python returns, the variables defined within the local scope of that function cease to exist. The local variables are destroyed. Function’s stack frame, which contains information about the function call, including local variables, is popped off the call stack.

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

Ans:

A "return value" in programming refers to the value that a function provides as its output when it's called. When a function is executed, it may perform certain computations or operations and then produce a result that it "returns" to the caller. This return value can be used for various purposes, such as storing data, performing further calculations, or making decisions based on the result.

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

Ans: If a function in Python does not have a return statement, or if it reaches the end of the function without executing a return statement, the function will implicitly return a special value called None. None is a built-in Python object that represents the absence of a value or a null value. It is often used to indicate that a function doesn't produce any meaningful result or that it has a side effect but doesn't return a specific value.

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

Ans: In Python, you can make a function variable refer to a global variable by using the global keyword within the function. This keyword allows you to indicate that a variable should be treated as a global variable, even if it has the same name as a local variable in the function.

10. What is the data type of None?

Ans: In Python, None is a built-in object that represents the absence of a value or a null value. It is often used to indicate that a variable or expression does not have a meaningful value. None itself is an instance of the NoneType data type.

11. What does the sentence import areallyourpetsnamederic do?

Ans: The sentence "import areallyourpetsnamederic" does not have any inherent meaning or functionality in Python because "areallyourpetsnamederic" is not a standard Python module or library. In Python, the import statement is used to import modules or libraries that provide additional functionality to your program.

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

Ans:

Code:

import spam

# Call the bacon() function from the spam module

spam.bacon()

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

Ans: To prevent a program from crashing when it encounters an error, you can implement error handling techniques. Error handling allows your program to gracefully handle unexpected situations or exceptions, which can help prevent crashes and make your code more robust.

1. Try-Except blocks
2. Exception Handling
3. Logging

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

Ans: In Python, the try and except clauses are used to implement error handling and exception handling mechanisms. They serve distinct purposes in handling exceptions.

Try: The try clause is used to enclose a block of code where you anticipate the possibility of an exception or error occurring

If an exception occurs within the try block, the execution of the try block is halted, and the control is transferred to the associated except block.

Except: The except clause follows the try block and specifies one or more exception types that it can handle. The purpose of the except clause is to define how your program should respond to specific types of exceptions that may occur during the execution of the try block.

You can have multiple except clauses to handle different types of exceptions.