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

**Ans.1** Functions are advantageous to have in programs because of the following reasons:-

1. **Readability and maintainability**

🡪Functions enhance the readability of your code by giving meaningful names to blocks of code.

1. **Abstraction**

🡪Functions encapsulate a set of operations or actions behind a single interface.

1. **Modularity**

🡪Functions allow you to break down complex tasks into smaller, manageable units of code.

1. **Code reuse**

🡪By defining functions, you can write a block of code once and reuse it multiple times throughout your program.

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

**Ans.2** The code inside a function runs when the function is called, not when it is specified or defined. Function definitions serve as a blueprint or template for the actions to be performed, but the code inside the function is executed only when the function is invoked or called using its name. The function call triggers the execution of the code block defined within the function body, and the program flow transfers to that function. Once the function completes its execution or reaches a return statement, the program flow returns to the point where the function was called, and the execution continues from there.

1. **What statement creates a function?**

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

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

**Ans.4**

. **Function:** A function is a block of code that performs a specific task or action. It is defined using the **def** statement, and it may or may not accept input parameters. Functions can have a return value to provide the result of their computation.

**. Function Call:** A function call is the act of invoking or executing a function. It is when you use the function name followed by parentheses, optionally passing any required arguments, to execute the code within the function. When a function is called, the program flow transfers to the function's code block, executes the statements within it, and then returns to the point where the function was called.

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

**Ans**.5 Python program generally has one global scope and multiple local scopes, depending on the number of functions and classes defined, including any nested functions or classes.

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

**Ans.6** When a function call returns, the local variables within the function's local scope cease to exist. They are no longer accessible or valid outside of the function. The memory allocated to the local variables is released, and their values are discarded.

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

**Ans.7** The concept of a return value in a function is the value that the function sends back to the caller after its execution. It is a way for a function to provide a result or output to the code that called it. A return value can be of any data type (such as integers, strings, lists, etc.), and it allows the caller to use and manipulate the result returned by the function. Yes, it is possible to have a return value in an expression. For example, you can use the return value directly in an assignment or as an argument to another function call.

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

**Ans.8** The concept of a return value in a function is the value that the function sends back to the caller after its execution. It is a way for a function to provide a result or output to the code that called it. A return value can be of any data type (such as integers, strings, lists, etc.), and it allows the caller to use and manipulate the result returned by the function. Yes, it is possible to have a return value in an expression. For example, you can use the return value directly in an assignment or as an argument to another function call.

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

**Ans.9**

global\_var = 10

def function():

global global\_var

print(global\_var)

function()

1. What is the data type of None?

**Ans.10** The data type of **None** in Python is **NoneType**. **None** is a special value in Python that represents the absence of a value or a null value. It is commonly used to indicate the absence of a meaningful result or to initialize variables when no specific value is assigned.

1. What does the sentence import areallyourpetsnamederic do?

**Ans.11** If "areallyourpetsnamederic" were an actual Python module or package, then the statement "import areallyourpetsnamederic" would import that module or package, making its functionality available in the current code file. However, without an existing module or package with that name, the statement has no effect and would result in a **ModuleNotFoundError** if executed.

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

**Ans.12** After importing the **spam** module, you can call the **bacon()** function using the module name **spam** as a prefix followed by the function name **bacon()**. This way, you can access and utilize the features provided by the **spam** module in your code.

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

**Ans.13**

try:

# Code that might cause an error

# ...

except ExceptionType:

# Code to handle the specific exception

# ...

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

**Ans.14**

**try clause:** The **try** clause is used to enclose the code that might raise an exception.

**except clause:** The **except** clause is used to specify how to handle a specific type of exception that occurred within the try block.