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

Sol:

Functions are advantageous to have in your programs for several reasons:

1. Code Reusability: Functions allow you to encapsulate a piece of code and reuse it multiple times throughout your program, reducing the amount of code you need to write and making your code easier to maintain.
2. Improved Organization: Functions help you organize your code into logical blocks, making it easier to understand and debug.
3. When does the code in a function run: when it&#39;s specified or when it&#39;s called?

Sol:

A function code runs when it's called, not when it's specified.

A function is a block of code that can be called by other parts of your program. When you specify a function, you define its name, parameters (if any), and the operations it should perform. The code within a function is not executed until you specifically call the function by using its name and passing any required arguments.

1. What statement creates a function?

Sol:

In most programming languages, the "function" or "def" keyword is used to create a function. Here is an example of how to create a function in Python:

def function\_name(arguments):

# code to be executed

return output

In this example, the keyword **def** is used to define a function named **function\_name** which takes in **arguments** as inputs and executes the code within the function. The **return** statement is used to specify the output of the function, which can be used by the code that calls the function.

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

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

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

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

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

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

10. What is the data type of None?

11. What does the sentence import areallyourpetsnamederic do?

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

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

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

Solutions:

1. A function is a block of organized, reusable code that is used to perform a single, related action. A function call is an expression that executes a function. When a function is called, the program control is transferred to the function definition.
2. In a Python program, there is only one global scope. However, there can be multiple local scopes within a program, one for each function definition.
3. Variables defined in a local scope are destroyed when the function call returns. This is because local variables are stored on the call stack, and when the function call is completed, the stack is popped, and the memory occupied by the local variables is freed.
4. A return value is a value that is returned by a function when it is finished executing. In Python, functions can return any type of value, including numbers, strings, lists, and objects. It is possible to have a return value in an expression, as long as the expression returns a value that can be used in the context where the expression is used.
5. If a function does not have a return statement, the return value of a call to that function is None.
6. To make a function variable refer to the global variable, you can use the global keyword in the function definition. For example, if you have a global variable named "x", you can refer to it in a function with the following code:

def my\_function():

global x

x = x + 1

1. The data type of None is NoneType.
2. The sentence **import areallyourpetsnamederic** will cause a **ModuleNotFoundError** because there is no module named "areallyourpetsnamederic" in the standard library or in any third-party packages.
3. If you have a **bacon()** feature in a **spam** module, you would call it **spam.bacon()** after importing **spam**.
4. To save a program from crashing if it encounters an error, you can use the **try** and **except** clauses in Python. The **try** clause specifies a block of code to be executed, and the **except** clause specifies a block of code to be executed if an error occurs in the **try** block.
5. The purpose of the **try** clause is to enclose a block of code that may raise an exception. The purpose of the **except** clause is to handle the exception that is raised in the **try** block, allowing the program to continue executing even if an error occurs.