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

Ans1

Reusability: Functions can be used repeatedly in a program, making it easier to write and maintain code. Instead of writing the same code multiple times, write a function once and call it whenever needed. This can save time and reduce the chances of errors in code.

Modularity: Functions help break down a program into smaller, more manageable pieces. Each function can handle a specific task or operation, making the overall program easier to understand and modify.

Encapsulation: Functions can hide the details of how they accomplish their task from the rest of the program. This makes it easier to change the implementation of a function without affecting other parts of the program.

Abstraction: Functions allows to think about a program in terms of higher-level concepts rather than low-level implementation details. By using well-named functions with clear inputs and outputs, make code more readable and easier to reason about.

Testing: Functions can be tested independently of the rest of the program. This makes it easier to ensure that each function works correctly and can help identify bugs more quickly.

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

Ans2

function does not execute the code inside it. Instead, it creates a function object and assigns it a name. When the function is called by using its name followed by parentheses, the code inside the function is executed.

1. What statement creates a function?

Ans3

def statement is used to create a function.

Eg

def add\_numbers(a, b):

"""This function adds two numbers together"""

return a + b

result = add\_numbers(2, 3)

print(result) # Output: 5

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

Ans4

A function is a block of code that performs a specific task and can be reused throughout a program. It is defined using the def keyword and consists of a function header (which includes the function name, parameters, and docstring) and a function body (which contains the statements that are executed when the function is called).

A function call, on the other hand, is the code that invokes the function and passes any required arguments to it. It consists of the function name followed by parentheses that contain any required arguments.

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

Ans5

In a Python program, there is exactly one global scope, which is created when the program starts executing. This global scope contains all the variables and functions defined at the top level of the program.

On the other hand, local scopes are created every time a function is called. Each function call creates a new local scope, which contains the variables and arguments specific to that function call. When the function call returns, the local scope is destroyed.

Therefore, the number of local scopes in a Python program is not fixed, as it depends on the number of function calls made during program execution. However, there is always exactly one global scope in a Python program.

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

Ans6

Local scope of that function is destroyed, and any variables defined within that local scope are also destroyed. This means that the variables defined in the local scope of a function cannot be accessed from outside the function.

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

Ans7

A return value is a value that a function can optionally return to the caller when it completes its task. When a function is called, it may perform some computations and then return a value, which can then be used by the caller. The return value can be of any data type, such as a number, string, list, or even another function.

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

Ans8

If a function does not have a return statement, the return value of a call to that function will be None.

In Python, if a function executes without encountering a return statement, it implicitly returns None at the end of the function. This means that the return value of the function is None by default.

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

Ans9

To make a function variable refer to the global variable, use the global keyword followed by the name of the global variable. This tells Python that the variable you are referencing in the function should be the global variable, rather than a local variable with the same name.

Eg

x = 10

def my\_function():

global x

x = 20

print("Inside function:", x)

my\_function()

print("Outside function:", x)

10. What is the data type of None?

Ans10

None is considered to be a data type of its own in Python and is often referred to as the NoneType. It is a singleton object, which means that there is only one instance of the None object in memory.

The data type of None can be checked by using the type() function.

11. What does the sentence import areallyourpetsnamederic do?

Ans11

The sentence "import areallyourpetsnamederic" is a syntactically valid Python statement, but it does not have any predefined meaning in Python. It is not a standard library or third-party module that can be imported and used in a Python program.

the statement "import areallyourpetsnamederic" does not refer to any valid module or package in Python, and therefore it will raise an error when executed in a Python program.

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

Ans12

bacon() feature in a spam module, you can call it after importing the spam module using the dot notation.

The dot notation is used to access the functions and variables defined in a module, and it follows the syntax module\_name.function\_name().

Therefore, to call the bacon() function from the spam module

import spam

spam.bacon()

In this code, import the spam module using the import statement, and then call the bacon() function using the dot notation. The spam.bacon() syntax tells Python to look for the bacon() function inside the spam module and execute it.

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

Ans13

Use exception handling: You can use try-except blocks to catch and handle exceptions that occur during the execution of your program. By wrapping potentially problematic code in a try block and specifying how to handle the exception in an except block, you can prevent the program from crashing and take appropriate action to handle the error.

Check for input validity: If your program takes user input, it's a good practice to check that the input is valid before performing any calculations or operations on it. This can help prevent errors that might occur due to incorrect input values.

Use default values: If your program relies on external data, such as data from a file or a database, you can use default values to ensure that your program can continue running even if the data is missing or incorrect. For example, you can provide default values for missing data or use fallback data sources if the primary source is unavailable.

Write defensive code: Writing defensive code involves anticipating potential errors or edge cases and adding safeguards to prevent them from causing a crash. For example, you can add input validation checks, range checks, and null checks to ensure that your code can handle unexpected input or data.

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

Ans14

The try and except clauses are used together to implement exception handling, which is a mechanism for detecting and responding to errors or exceptional conditions in a program.

The try clause is used to define a block of code that might raise an exception. The code in the try block is executed normally, but if an exception is raised during the execution, control is transferred to the except clause.

The purpose of the try clause is to attempt to execute a piece of code that may raise an exception and provide a way to handle the exception if it occurs. This allows the program to gracefully handle errors and continue running, rather than crashing with a traceback.