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

* Promote code reuse.
* Enhance readability and organization.
* Simplify debugging and maintenance.
* Allow modular programming.
* Enable abstraction and encapsulation.
* Facilitate collaboration and code sharing.

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

* The code in a function runs when it is called, not when it is specified.

1. What statement creates a function?

def my\_function():

pass

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

* A function is a block of code that performs a specific task and is defined using a def statement. A function call is the code that executes the function, typically by using the function's name followed by parentheses.

Function definition:

def my\_function():

print("Hello, world!")

Function call:

my\_function()

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

* In a Python program, there is only one global scope. However, there can be multiple local scopes, which are created whenever a function is called. Each function call creates its own local scope.

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

* When a function call returns, the local scope of that function is destroyed, and all variables within that local scope are discarded. This means the local variables cease to exist and their values are no longer accessible.

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

* The concept of a return value refers to the value that a function sends back to the caller after it finishes executing. This value can be used in the place where the function was called. Yes, it is possible to use 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?

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

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

* global keyword
* Using global x inside the function tells Python that x refers to the global variable x defined outside of the function, rather than creating a new local variable x inside the function. This way, changes made to x within the function will affect the global variable x.

10. What is the data type of None?

* The data type of None in Python is NoneType.

11. What does the sentence import areallyourpetsnamederic do?

* The sentence import areallyourpetsnamederic in Python is syntactically valid, but it doesn't import any standard Python module or package because areallyourpetsnamederic is not a built-in module or a commonly used third-party module.
* Python's import statement is used to import modules or packages that provide additional functionality. If areallyourpetsnamederic were to refer to a module or package that exists in your Python environment, it would import that module or package, making its functions, classes, or variables accessible in your current Python script. However, using unconventional or non-existent names like areallyourpetsnamederic would typically result in an ImportError unless you have a module with that name defined in your project.

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

* Here, spam is the module name, and bacon() is the function defined within the spam module. Using spam.bacon() calls the bacon() function and executes its code.
* import spam
* spam.bacon()

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

* To prevent a program from crashing when it encounters an error, you can use exception handling. In Python, this is done using `try` and `except` blocks. Here's how you can do it:

try:

result = 10 / 0 # This will raise a ZeroDivisionError

except ZeroDivisionError:

print("Error: Division by zero!")

except:

print("An error occurred!")

In this example:

- The code inside the `try` block is attempted.

- If an error occurs (in this case, a `ZeroDivisionError`), Python jumps to the appropriate `except` block.

- If no specific `except` block matches the error, it falls back to the general `except` block.

- Using exception handling allows you to gracefully handle errors and continue executing the program rather than crashing. You can also log errors, notify users, or take other appropriate actions based on the type of error encountered.

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

* **Purpose of the try clause:** The try clause is used to enclose the code that you anticipate might raise an exception or error. It allows you to test a block of code for errors. If an error occurs within the try block, Python looks for an appropriate except block to handle the error.
* **Purpose of the except clause:** The except clause is used to handle specific exceptions that occur within the try block. When an exception is raised that matches one of the except clauses, Python executes the code within that except block. This allows you to handle errors gracefully and continue with the program execution even if errors occur.