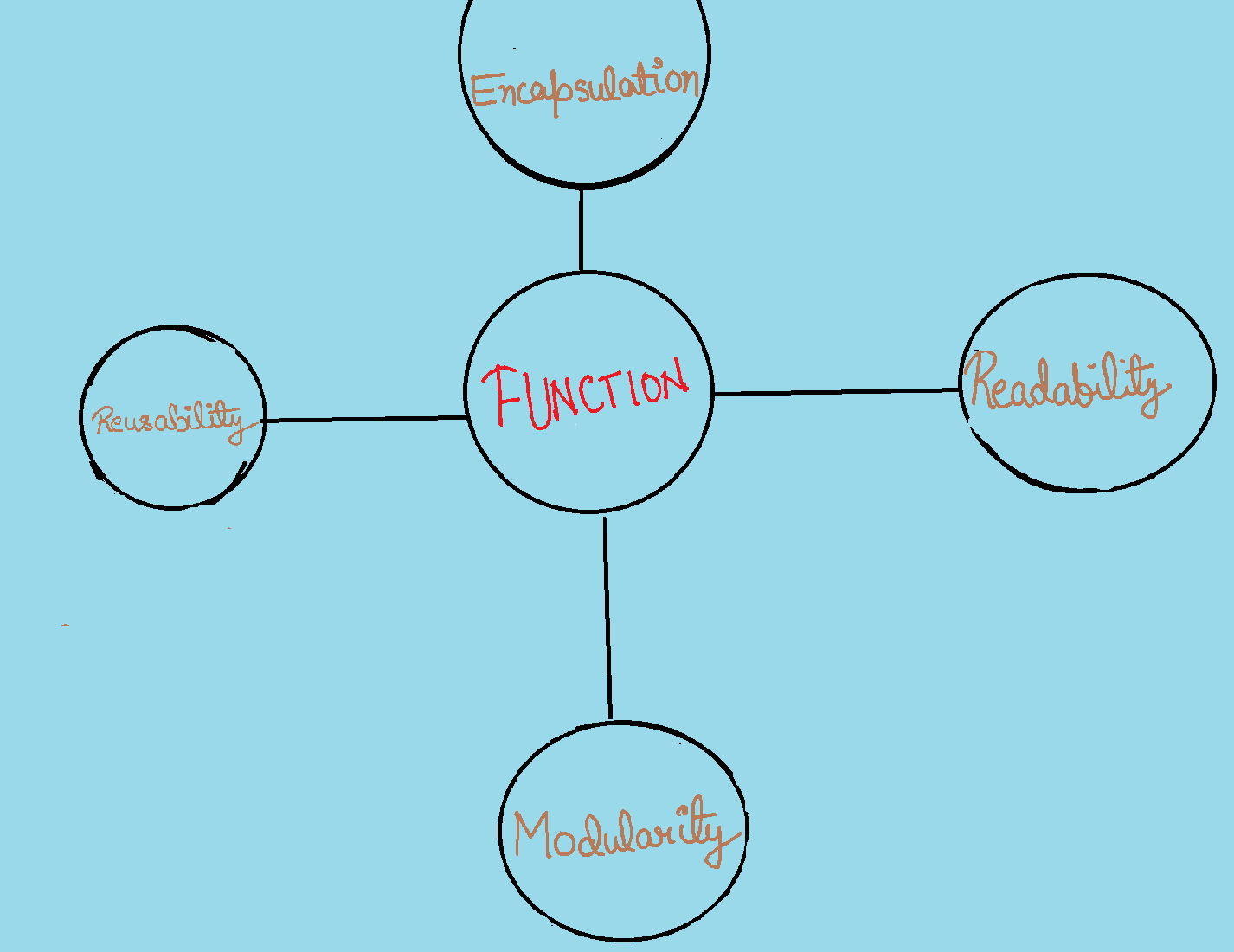
ASSIGNMENT 03

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

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

****

**1*. Modularity: Functions allow you to break down your code into smaller, manageable, and reusable pieces. Each function can perform a specific task or solve a particular problem, making your code more organized and easier to understand.***

**2. *Reusability: Once you define a function, you can call it multiple times from different parts of your program. This promotes code reuse, reducing the need to duplicate code and making maintenance and updates more efficient.***

**3. *Readability: Well-named functions make your code more readable and self-documenting. Function names should describe what the function does, making it easier for others (and your future self) to understand your code.***

**4. *Testing and Debugging: Functions make it easier to test and debug your code. You can isolate individual functions and test them independently, which simplifies the process of identifying and fixing issues.***

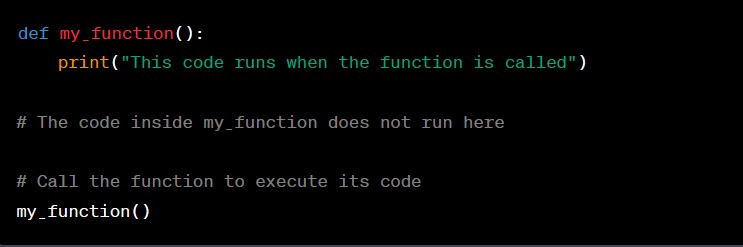
**5. *Encapsulation: Functions provide a level of encapsulation, allowing you to hide the internal details of a task. This can be especially useful for creating libraries and APIs where users only need to interact with well-defined functions.***

**6. *Code Organization: Functions help you organize your code logically. You can group related functionality into separate functions, making it easier to locate and maintain specific parts of your program.***

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

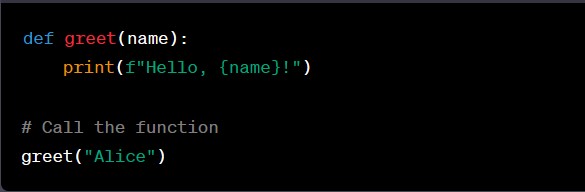
**Ans***: In Python, the code in a function runs when the function is called, not when it's specified or defined.*

*When you define a function in Python, you are essentially creating a reusable block of code that will be executed when the function is invoked elsewhere in your program. The code inside the function is only executed when you call the function using its name followed by parentheses. Here's an example:*

****

*In this Python example, the code inside `my\_function` does not run when the function is defined. It only runs when you call `my\_function()`, which will print the specified message to the console.*

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

**Ans:** *In Python, the “def” statement is used to create or define a function. In Python, the basic syntax for defining a function is as follows***:** **

*In this example, the def statement is used to create a function called greet, which takes a parameter name. The code inside the function defines how the greeting message is printed. When you call greet("Alice"), it will execute the code within the greet function and print "Hello, Alice!" to the console.*

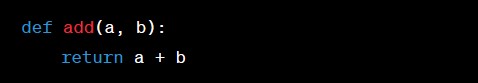
1. **What is the difference between a function and a function call?**

**Ans:**

**1. Function:**

* *A function is a block of code that performs a specific task or a set of tasks.*
* *It is defined with a name, and it may accept input values known as parameters or arguments.*
* *Functions are typically defined once in your code and can be used multiple times throughout your program.*
* *Functions are declared with the `def` keyword in Python and other similar keywords in different programming languages.*

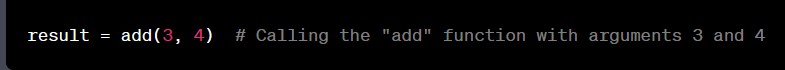
**Example:**

****

**2. Function Call:**

* ***A function call is the act of invoking or executing a function.***
* ***When you call a function, you provide the required arguments (if any) and execute the code inside the function.***
* ***Function calls are used to utilize the functionality provided by a function at a specific point in your program.***

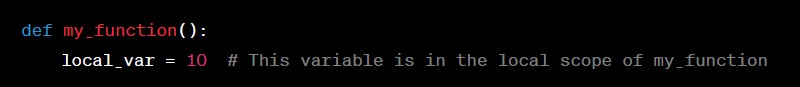
**Example:**

****

*In summary, a function is a defined block of code with a specific purpose, while a function call is the action of actually running that code with specified inputs.*

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

**Ans:** *In a Python program, there is typically one global scope and multiple local scopes.*

* **Local Scopes:**
* *There can be multiple local scopes in a Python program, one for each function or code block.*
* *Local scopes are created whenever you define a function or enter a code block (e.g., inside loops or conditional statements).*
* *Variables defined within a local scope are only accessible within that scope****.***
* ***Example of a local variable within a function:***
* ******

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

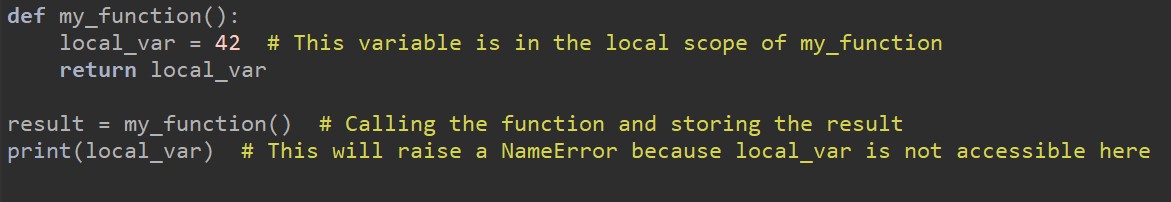
***Ans****:*

*When a function call in Python returns, the local variables that were defined within the function's scope cease to exist. This process is known as variable scope and lifetime. Here's what happens to variables in a local scope when a function call returns:*

***1.*** *Variable Lifetime: Local variables are created when a function is called and are destroyed (their memory is released) when the function call completes, either by reaching the end of the function or encountering a `return` statement.*

**2.** Inaccessibility*: Once the function call returns, you cannot access the local variables defined within that function from outside the function. Attempting to access them will typically result in a `Name Error` because the variables are no longer in scope*.

Here's an example to illustrate this



*In this example, `local\_var` is a local variable defined within the `my\_function` function. When the function is called and completes, the local variable `local\_var` goes out of scope, and you cannot access it outside the function.*

*It's important to note that if you want to use the value of a local variable outside the function, you should either return that value from the function and store it in a global variable or pass it as a return value and assign it to a variable in the calling code.*

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

Ans: *The concept of a return value in programming pertains to the value that a function provides as output when it is called and executed. In most programming languages, including Python, functions can return data or values to the caller using the return statement. The return value is the result or data that the function produces and sends back to the point in the code where the function was called.*

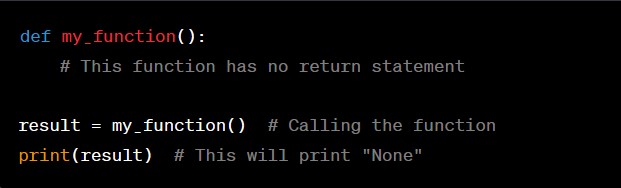
*Yes, it is possible to use a return value in an expression. The return value from a function can be used just like any other value in your code. You can perform operations on it, assign it to variables, use it in conditional statements, and more.*

*For example:*

*In this case, the result variable holds the return value of the add function, and it is used in an expression to calculate a new value stored in the total variable.*

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

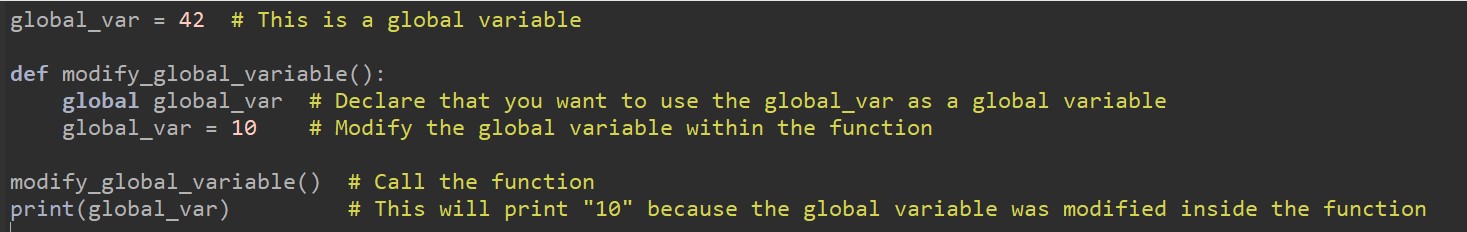
**Ans:** *If a function in Python does not have a `return` statement, it implicitly returns a special value called `None`. `None` represents the absence of a value or the lack of a return value. When you call a function without a `return` statement, it still returns `None` by default.* **Here's an example:**

**

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

Ans: *To make a function variable refer to a global variable in Python, you can use the `global` keyword within the function to declare that a particular variable should be treated as a global variable, rather than a local variable within the function's scope. This allows you to access and modify the global variable from within the function.*

*Here is the Example:*

**

10. What is the data type of None?

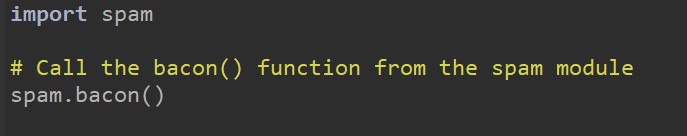
Ans: *In Python, `None` is a special value that represents the absence of a value or the lack of a return value. It is often used to indicate that a variable or expression doesn't have a meaningful value or that a function doesn't return a specific result. `None` itself is of the data type `NoneType`.*

**11. What does the sentence import areallyourpetsnamederic do?**

**Ans:** *The sentence "import areallyourpetsnamederic" is not a valid Python import statement, and it will result in a ModuleNotFoundError if you try to execute it in Python.*

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

Ans:

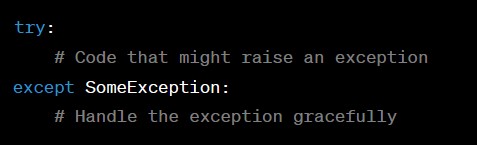


*In Python, when you import a module using import, you can access the functions and variables defined in that module by prefixing them with the module name and a dot (.) separator. This way, you can avoid naming conflicts and clearly specify which module the function or variable belongs to.*

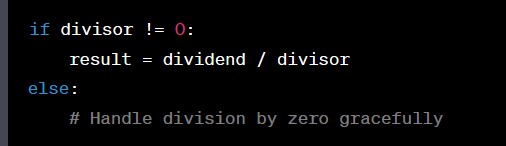
**13. What can you do to save a programme from crashing if it encounters an error?**

**Ans:** *To prevent a program from crashing when it encounters an error, you can use various error-handling techniques and strategies in your code. These techniques help you gracefully handle errors or exceptions that might occur during program execution. Here are some common approaches:*

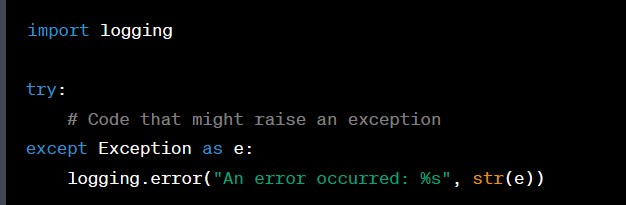
**1***.Try-Except Blocks (Exception Handling):**Use `try` and `except` blocks to catch and handle exceptions. This allows you to specify code that should run when an exception occurs, preventing the program from crashing.*

****

**2.** *Check for Errors:**Before performing potentially risky operations, check conditions or validate inputs to ensure they are within acceptable ranges or formats. If they're not, handle the situation appropriately.*

****

**3***.Logging:**Implement logging to record errors and relevant information. Logging allows you to capture error details without disrupting the program's flow.*

****

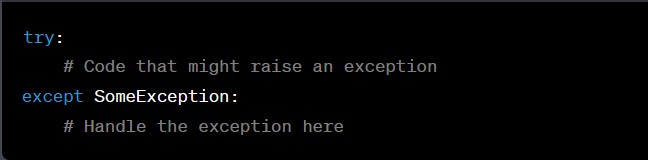
**14. What is the purpose of the try clause? What is the purpose of the except clause?**

**Ans:** **In Python, the `try` and `except` clauses are used together to handle exceptions or errors that might occur during the execution of a block of code. They serve the following purposes:**

**1.***Purpose of the `try` Clause***:**

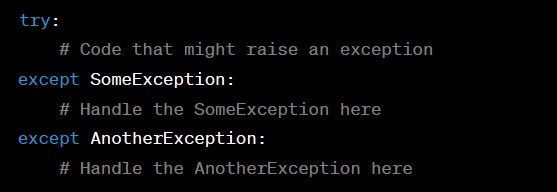
* + *The `try` clause is used to enclose the code that might raise an exception.*
  + *It allows you to specify a block of code where you anticipate that exceptions may occur.*
  + *If an exception occurs within the `try` block, the normal flow of the program is interrupted, and the program starts looking for an appropriate `except` block to handle the exception.*

**Example:**

****

**2.** *Purpose of the `except` Clause***:**

* + *The `except` clause follows the `try` block and is used to define how to handle specific exceptions or error conditions.*
  + *It specifies the type of exception it can catch, and if an exception of that type occurs within the associated `try` block, the code within the corresponding `except` block is executed.*
  + *You can have multiple `except` blocks to handle different types of exceptions.*
  + ***Example:***

****

*In summary, the `try` clause is used to encapsulate code where exceptions are expected, and the `except` clause(s) provide a way to gracefully handle those exceptions, allowing the program to continue running or providing error information to the user. Using `try` and `except` together is a fundamental aspect of error handling and exception management in Python.*