**Assignment 4**

**Python Basics**

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

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

* **Modularity**: Functions allow you to break your code into smaller, reusable pieces, making your code more organized and easier to maintain.
* **Reusability**: You can call a function multiple times in your program, avoiding code duplication and saving time.
* **Abstraction**: Functions provide a way to abstract away complex operations, making your code more readable and understandable.
* **Testing**: Functions make it easier to test individual parts of your code independently.

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

**Answer-** The code in a function runs when it's called. Functions are defined with the def statement, but they don't execute their code until they are invoked or called.

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

**Answer-** The **def** statement is used to create a function. Here's the syntax:

def function\_name(parameters):

Function code here

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

**Answer-**A function is a block of code that defines a specific task or operation. It contains the code to perform that task. A function call, on the other hand, is the act of invoking or executing the function to perform the specified task.

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

**Answer-**In a Python program, there is one global scope. Global variables are defined at this level. Local scopes are created whenever a function is called, and they exist only within the function. Each function call creates a new local scope.

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

Answer- When the function call returns, the variables in its local scope cease to exist. They are destroyed, and their values are no longer accessible.

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

**Answer-** The concept of a return value is that a function can provide a result or value back to the caller. Yes, it is possible to have a return value in an expression, and you can use the return value in further computations.

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

**Answer-** If a function does not have a return statement, it returns **None** by default. **None** is a special Python value that represents the absence of a value.

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

Answer-To make a function variable refer to a global variable, you can use the **global** keyword within the function. For example:

def my\_function():

global my\_variable

my\_variable = 41

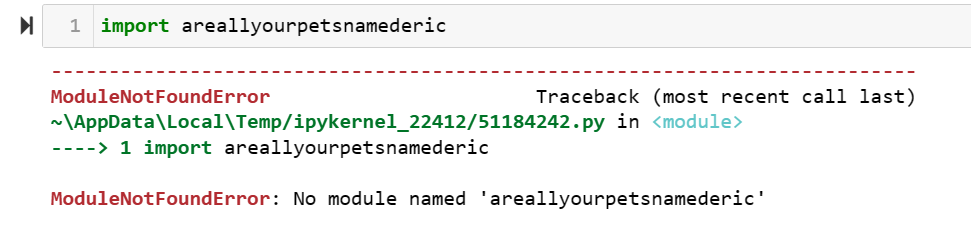
1. **What is the data type of None?**

Answer- The data type of **None** is **NoneType**.

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

Answer-The sentence **import areallyourpetsnamederic** attempts to import a Python module named **areallyourpetsnamederic**. If such a module exists and can be imported, it will become available for use in the current program.

But as its not available as of now so output will give error-



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

**Answer-** If you have a bacon() feature in a spam module, you can call it after importing spam like this:

import spam

result = spam.bacon()

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

**Answer-** To save a program from crashing when it encounters an error, you can use error handling techniques such as try and except blocks to handle exceptions gracefully. This allows your program to continue executing even if it encounters errors.

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

**Answer-** The try clause is used to enclose code that might raise an exception. It allows you to catch and handle exceptions gracefully. The **except** clause specifies what to do when a specific exception is raised. It is used to handle exceptions and provide alternative code paths in case of errors.