# Assignment 3

### **Question 1:-**

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

#### **Ans:-**

Function reduces the need for duplicate code. This makes programs shorter, easier to read, easier to read, and easier to update.

The use of functions **helps avoid duplication of code and effort**. As you develop a program, it may be necessary to perform the same tasks multiple times. In these situations using a function can reduce code duplication and manual effort. It reduces the size of both the source code and executable code. As a result, developing, testing, debugging, and maintaining services takes less time, which reduces the cost of development.

### **Question 2:-**

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

#### **Ans:-**

The code in a function executes when the function is called, not when the function is defined.

When a function is “called” the program “leaves” the current section of code and begins to execute the first line inside the function, thus the function “flow of control” is:

1. The program comes to a line of code containing a “function call”.
2. The program enters the function (starts at the first line in the function code).
3. **All instructions** inside the function are executed from top to bottom.
4. The program leaves the function **and returns to where it started.**
5. Any data computed and **RETURNED** by the function is used in place of the function in the original line of code.

### **Question 3:-**

**What statement creates a function?**

#### **Ans:-**

The def statement defines (that is, creates) a function.

A function is created with the def keyword. The statement in the block of the function must be indented.

Def function():

pass

The def keyword is followed by the function name with round brackets and a colon. The indented statement forms the body of the function.

The function is later executed when needed. We say that we call functions. If we call a function, the statement inside the function body is executed. They are not executed until the function is called.

### **Question 4:-**

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

#### **Ans:-**

A function consists of the def statement and the code in its def clause.

A function call is what moves the program execution into the function, and the function call evaluates the function’s return value.

| **Function** | **Function call** |
| --- | --- |
| A Function is a block of code that only runs when it is called. | Once a function is created in Python, we can call it by writing function\_name() itself. |
| There are two types of Functions   1. User-defined Function 2. Built-in Function | Types of Function call   1. Call by Value 2. Call by Reference |
| Example:-  # A simple Python function  def fun():  print("Welcome to GFG")  # Driver code to call a function  fun()  Output:-  Welcome to GFG | Example:-  def my\_function():  print("Hello from a function")  my\_function()  Output:-  Hello from a function |

### **Question 5:-**

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

#### **Ans:-**

There is one global scope, and a local scope is created whenever a function is called.

There is only one global Python scope per program execution. This scope remains in existence until the program terminates and all its names are forgotten. Otherwise, the next time you were to run the program, the names would remember their values from the previous run.

The local scope or function scope is a Python scope created at function calls. At any given time during execution, you’ll have at most four active Python scopes — local, enclosing, global, and built-in — depending on where you are in the code.

### **Question 6:-**

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

#### **Ans:-**

When a function returns, the local scope is destroyed, and all the variables in it are forgotten.

When the execution of the function terminates (returns), the local variables are destroyed. Codeless helps you visualize this because the local variables disappear after the function returns.

### 

### **Question 7:-**

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

#### **Ans:-**

A return value is a value that a function call evaluates. Like any value, a return value can be used as part of an expression.

In general, a function takes arguments (if any), performs some operations, and returns a value (or object). The value that a function returns to the caller is generally known as the function’s return value. All Python functions have a return value, either explicit or implicit.

A Python function will always have a return value. There is no notion of procedure or routine in Python. So, if you don’t explicitly use a return value in a return statement, or if you totally omit the return statement, then Python will implicitly return a default value for you.

### **Question 8:-**

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

#### **Ans:-**

If there is no return statement for a function, its return value is None.

If no return statement appears in a function definition, control automatically returns to the calling function after the last statement of the called function is executed. In this case, the return value of the called function is undefined.

### 

### **Question 9:-**

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

#### **Ans:-**

A global statement will force a variable in a function to refer to the global variable.

Variables that are created outside of a function are known as global variables.

Global variables can be used by everyone, both inside of functions and outside.

Normally, when you create a variable inside a function, that variable is local, and can only be used inside that function.

To create a global variable inside a function, you can use the global keyword.

### 

### **Question 10:-**

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

#### **Ans:-**

The data type of None is NoneType.

The None keyword is used to define a null value or no value at all.

None is not same as 0, False, or an empty string. None is a data type of its own (NoneType) and only None can be None.

### 

### **Question 11:-**

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

#### **Ans:-**

That import statement imports a module named areallyourpetsnamederic.

### 

### **Question 12:-**

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

#### **Ans:-**

This Function can be called with spam.bacon().

### 

### **Question 13:-**

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

#### **Ans:-**

Place the line of code that might cause an error in a try clause.

When it encounters an error, the control is passed to the except block, skipping the code in between. As seen in the above code, we have moved our code inside a try and except statement. Try running the program and it should throw an error message instead of crashing the program.

### 

### **Question 14:-**

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

#### **Ans:-**

The code that could potentially cause an error goes in the try clause.

The code that executes if an error happens goes in the except clause.

The Try and Except statement is used to handle these errors within our code in Python. The try block is used to check some code for errors i.e. the code inside the try block will execute when there is no error in the program. Whereas the code inside the except block will execute whenever the program encounters some error in the preceding try block.