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

**Answer:** Functions reduce the need for duplicate code. This makes programs shorter, easier to read, and easier to update. A Function call is what moves the program execution into the function, and the function call evaluates to the function’s return value.

2. When does the code in a function execute: when the function is defined or when the function is called?

**Answer:** The Code in a function executes when the function is called, not when the function is defined.

3. What statement creates a function?

**Answer:** The “def” keyword is a statement for defining a function in python. You will start a function using “def” keyword, specify a name followed by a colon(:) sign. The “def” call creates the function object and assigns it to the name given. You can further re-assign the same function object to other names.

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

**Answer:** A **FUCNTION** is a block of code that does a particular operation and returns a result. It usually accepts inputs as parameters and returns a result. The parameters are not mandatory. A **FUNCTION CALL**  is the code used to pass control to **FUCNTION.**

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

**Answer:** There’s only one **global Python scope** per **program** execution. This **scope** remains in existence until the **program** terminates and all its name are forgotten. Otherwise, the name time you were to run the **program,** the names would remember **their** values from the pervious run.

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

**Answer:** By default, the assignment statement creates **variables**  in the **local scope.** So the assignment inside the **function** does not modify the global **variable** a, it creates a new **local variable called** a, and assigns the value 3 to the **variable.**

7. What is a return value? Can a return value be part of an expression?

**Answer:** A return statement is used to end the execution of the function call and “returns” the result to the caller. The statements after the **return statements** are not executed. If the return statement is without any expression, then the special value None is returned.

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

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

9. How can you force a variable in a function to refer to the global variable?

**Answer:** If you want to **refer** to a **global variable in a function,** you can use the **global** keyword to **declare** which **variables** are **global.**

10. What is the data type of None?

**Answer:** The **None** keyword is used to define a null value, or no value at all. **None** is not the same as 0, False, or an empty string. **None** is a **data type** of its own (None Type) and only **None** can be **None.**

11. What does the import areallyourpetsnamederic statement do?

**Answer:** That **import statement imports a module** named **areallyourpetsnamederic.** This function **can** be called with spam.

12. If you had a function named bacon() in a module named spam, how would you call it after importing spam?

**Answer:** using dot notation

spam.bacon()

13. How can you prevent a program from crashing when it gets an error?

**Answer:** Instead, **error** handling can be used to notify the user of why the **error** occurred and gracefully exit the process that caused the **error.**

14. What goes in the try clause? What goes in the except clause?

**Answer:** A **try clause** is executed up until the point where the first exception is encountered. Inside the **except clause**, or the exception handler, you determine how the program responds to the exception. You can anticipate multiple exceptions and differentiate how the program should respond to them.