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

**Answer 1:** Functions saves a lot of time of our time. Whenever any operation is to perform multiple times so instead of writing the complete code again and again, we can just simply create a function. Then this function can be called any times, saving both time and line of codes.

For example: There is a sum operation to be used multiple times, so better to make a sum function and use it multiple times.

def sum1():

a = float(input(“Enter 1st number”))

b = float(input(“Enter 2nd number”))

return a+b

Now this sum function can be called as per the requirement in the code.

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

**Answer 2:** A function run when it is called. It won’t work when it is specified.

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

**Answer 3:** A function is created using def function. For example:

def function\_name():

#code stuff

return something

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

**Answer 4:**

Function: A function is a set of reusable code that performs for a specific task. It is used multiple times for the same function. A function is created using a def function and returns something.

Function call: It is the phase/time when the function is called to perform any action.

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

**Answer 5:** In Python, there is typically one global scope. This global scope includes variables and functions defined at the top level of the program or module, making them accessible throughout the entire program.

Local scopes, on the other hand, can be created within functions or blocks of code using constructs like functions, loops, or conditional statements. Each time a function is called, a new local scope is created for that function.

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

**Answer 6:** Variables in a local scope are temporary and only exist for the duration of the function's execution. Once the function finishes executing and returns a value or completes its task, the local scope is discarded, and the variables within it are no longer accessible from other parts of the program.

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

**Answer 7:** The concept of a return value means the output it produces after the execution of function. When a function is called, it will perform any task as per the code and as output it will give return statement. For example:

def function\_name():

#perform\_some\_operation

return something

Yes, it is possible to have return value in an expression. For example:

result1 = function\_name(3,4)

final\_result = result1 \* 2

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

**Answer 8:** It will return None which does not have any value.

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

**Answer 9:** To make a function variable refer to a global variable in Python, you can use the global keyword within the function. This informs Python that you want to use the global variable instead of creating a new local variable with the same name. Here's an example:

global\_variable = 10

def use\_global\_variable():

global global\_variable

print(global\_variable)

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

**Answer 10:** Data type of None is NoneType. We can check this with below code syntax.

print(type(None))

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

**Answer 11:** It will give an error as there is not any module with name – *areallyourpetsnamederic*. We can only import modules which are available in Python.

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

**Answer 12:** After importing spam, we can call the bacon function easily like below syntax.

import spam

spam.bacon()

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

**Answer 13:** To save a programme from crashing if it encounters an error, we can use the try and except block. Using try and except block, we can not only prevent unexpected error but also gets the type of error we are getting and can also put a statement which may come as result when program crashes.

For example:

try:

output= 10/0

print(output)

except ZeroDivisionError:

print(“There is a zero division error in this code!”)

except Exception as e:

print(f“There is a {e} type of error in above code!”)

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

**Answer 14:**

Purpose of try clause: Try clause encloses the code that may causes an error or multiple errors. If any error occurs due to the try clause code, Python will stop this block execution and will move to except block automatically.

Purpose of except clause: Except clauses in a way handles the error occurred with the try block codes. It reflects the type or error as we write the code. We can create multiple except block as multiple errors may occurs.