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

* Reusability
* Abstraction
* Modularity
* Organization
* Debugging

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

* The code in a function runs when the function is called, not when it's defined

3. What statement creates a function?

* The ‘def’ statement

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

* The function is a block of reusable code that performs tasks and that can be called using ‘def’.
* A function call is an instruction that tells the program to execute a particular function

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

* There is one global scope, which is created when the program starts and exists until the program terminates.
* Local scopes are created whenever a function is called, and they exist only for the duration of the function call.

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

* When a function call returns, its local scope is destroyed along with any variables, functions, or objects defined within that function.

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

* When a function is called, it performs some operation and then returns a value to the caller. The return value can be a single value or a collection of values, and it can be of any data type. Yes, it is possible to have a return value in an expression.
* For ex:

def add\_numbers(a, b):

return a + b

result = add\_numbers(2, 4) \* 4

print(result)

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

* ‘None’

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

* When we use the **global** keyword with a variable name inside a function, it tells Python to look for the variable in the global scope instead of creating a new local variable.
* For ex:

x = 11 # global variable

def foo():

global x

x=6 #update the global variable

foo()

print(x) # Output: 6

10. What is the data type of None?

* NoneType

11. What does the sentence import areallyourpetsnamederic do?

* **‘ModuleNotFoundError’**

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

🡪 import spam

spam.bacon()

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

* Try-except blocks can be used

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

* The **try** clause allows us to write code that may throw an exception, while the **except** clause provides a way to handle the exception and keep the program from crashing.