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

Ans.)  **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 run: when it's specified or when it's called?

Ans.)The code in a function run when it’s called.

3. What statement creates a function?

Ans.) A function in Python is a logical unit of code containing a sequence of statements indented under a name given using the “def” keyword.

def function\_name(parameters):

This above statement creates a function.

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

Ans.) A function 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.  
  
E.g:  
Function add(a,b)  
return a+ b  
  
A function call is the code used to pass control to a function.  
  
E.g.:  
  
b = add(5,6)  
  
Now b will have the value 11.

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

Ans.) **There's** 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.

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

Ans.)The local variable can be used outside the function any time after the function call completes.

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

Ans.) A **return** statement is used to end the execution of the function call and “**returns**” the result (**value** of the **expression** following the **return** keyword) 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?

Ans.) **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.

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

Ans.) 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?

Ans.) The **None** keyword is used to define a **null variable** or an object. In **Python**, **None** keyword is an object, and it is a **data type** of the class NoneType . We can assign **None** to any **variable**, but you can not create other NoneType objects.

11. What does the sentence import areallyourpetsnamederic do?

Ans.) That import statement imports a module named areallyourpetsnamederic.

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()**.

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

Ans.) If an error occurs in a program, we don’t want the program to unexpectedly crash on the user. 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 is the purpose of the try clause? What is the purpose of the except clause?

Ans.) The **try block** lets you test a **block** of code for errors. The except **block** lets you handle the error. The finally **block** lets you execute code, regardless of the result of the **try**- and except blocks.