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

*Function enables us to bundle a set of instructions which we can use repeatedly. Is is a reusable piece of code that can be shared and used by others. Functions break long programs up into smaller components and make them more maintainable and modular.*

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

*Every function has a name (also called function identifier) by which it can be called. When the function is called, the code in it runs. For example z = pow(2,3) - here we are calling pow function by passing arguments 2 and 3 to get the value of 2 raised to the 3 power, i.e. 8 and then assigning the return value to a variable z.*

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

*def func\_name():*

*# specify what the function will do*

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

*Function is created using def keyword along with a unique identifier as its name followed by parentheses and a colon as mentioned below:*

*def subtractNum():*

*print(34 - 4)*

*Function call is made by writing the name of the function followed by parentheses:*

*subtractNum()*

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

*There's only one global scope per program execution*

*Local scope is defined inside a function. So the number of local scopes depends on how many functions are defined in the whole program.*

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

*The variables are destroyed - as they are local their lifetime is spanned along the timeline of the function execution.*

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

*Return values are those objects which are sent by the functions back to their callers.All Python functions have a return value, either explicit or implicit. When a return statement is used inside a function the value following the return keyword is the return value. For example,*

*def do\_sum():*

*a=2*

*b=3*

*c=a+b*

*return c*

*x=do\_sum()*

*print("The returned value from the function do\_sum() is ",x)*

*The return value of a Python function can be any Python object. Everything in Python is an object. So, your functions can return numeric values (int, float, and complex values), collections and sequences of objects (list, tuple, dictionary, or set objects), user-defined objects, classes, functions, and even modules or packages.*

*The return statement can be defined inside function and it an return value from expression:*

*def do\_sub():*

*return 3-2*

*y=do\_sub()*

*print("The returned value from the function do\_sub() is ",y)*

**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?**

*By using global keyword to declare which variables are global. For example:*

*myVar=1*

*def acc\_glob():*

*global myVar*

*myVar =1.5*

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

*The None keyword is used to define a null value or null object, or no value at all. It is not the same as 0, False, or an empty string. None is a data type of its class NoneType and only None can be one instance of NoneType*

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

*It makes the code of the module <areallyourpetsnamederic> available to another where this statement is used.*

**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?**

*Exception Handling and Logging*

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

*Python would process all code inside the try and except statement. When it encounters an error, the control is passed to the except block, skipping the code in between.*