1. Why are functions advantageous to have in your programs?
   Solution 1: A function can be defined as a collection of statements that performs a given task. Functions are advantageous because:
   - Functions mitigate redundancy or duplication.
   - They reduce length of a program hence, making it easier to read and update.
   - Are reusable
   - Maintains modular structure of a program.

2. When does the code in a function run: when it's specified or when it's called?
   Solution 2: The code in a function runs when it is called. It does not run when it is specified or defined. For e.g.:
   ```
   def calc(x):
           x= x*x
           print(x)
   calc(10)        #it will run when called
   ```

3. What statement creates a function?
   Solution 3: We use def to define/ create a function. For e.g. def calc().

4. What is the difference between a function and a function call?
   Solution 4: A function is procedure to achieve a particular result while function call is using this function to achieve that task and evaluates the return value of the function.

5. How many global scopes are there in a Python program? How many local scopes?
   Solution 5: One local as well as one global scope is created whenever a function is called.

6. What happens to variables in a local scope when the function call returns?
   Solution 6: A function's local variables exist only while function is executing i.e. when a functions ends all its variables and parameters gets destroyed.

7. What is the concept of a return value? Is it possible to have a return value in an expression?
   Solution 7: A return is a value that a function returns to the calling script or function when it completes its task. A return value can be any one of the four variable types: handle, integer, object, or string.
   - 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.
   - A return statement is overall used to invoke a function so that the passed statements can be executed.

   For e.g. :
   ```
   x=2
   def pro(x):
     return(x*3)
   y =pro(x)
   print(y)
   ```

8. If a function does not have a return statement, what is the return value of a call to that function?
   Solution 8: The return value will None.

9. How do you make a function variable refer to the global variable?
   Solution 9: We use 'global' keyword to make a function variable refer to the global variable.

10. What is the data type of None?
    Solution 10: The data type of None is: NoneType.

11. What does the sentence import areallyourpetsnamederic do?
    Solution 11. If a python module really exists then, by this way it will import the module with that name. Here, areallyourpetsnamederic do will get imported but since there is no such module then we will get an error.

12. If you had a bacon() feature in a spam module, what would you call it after importing spam?
    Solution 12: We call call the function along with it's module name like: spam.bacon().

13. What can you do to save a programme from crashing if it encounters an error?
    Solution 13: The code that may cause an error can be placed inside try clause to save the programme from crashing if it encounters an error.

14. What is the purpose of the try clause? What is the purpose of the except clause?
    Solution 14: Both the clauses are executed to handle error in a program. We put that part of our code under try clause which can give an error whereas, after getting an error which code should be executed is put under except clause.
    For e.g.:

```
x= 8
y=0
def divide(x, y):
    try:
        result = x // y
        print("Yeah ! Your answer is :", result)
    except ZeroDivisionError:
        print("Sorry ! You are dividing by zero ")
```