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

Answer: Following are the reasons that functions are advantageous to have in program:

1. Modularity: Functions allow you to break down complex problems into smaller, manageable parts that can be written, tested, and debugged independently.

2. Reusability: Functions can be called multiple times within a program, reducing the need to write the same code repeatedly, saving time and making the code more maintainable.

3. Abstraction: Functions allow you to hide the implementation details of a certain behaviour and expose only the necessary information to the rest of the program.

4. Improved readability: By breaking down complex logic into smaller, named functions, the overall code becomes easier to read and understand.

5. Easier debugging: If a function is causing an error, it's easier to isolate and fix because the error is contained within the function, rather than scattered throughout the program.

6. Testing: Functions can be easily tested in isolation, improving the reliability and quality of the code.

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

Answer: The code in a function is ran when the function is called explicitly.

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

Answer: The syntax to create a function is as follows:

```
Def func_name():

        #Function Body
```

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

Answer: The difference between a function and a function call is as follows:

1) Function: A function is a block of code that performs a specific task. It has a name consists of parameters and a return value if required. Functions can be defined once and then called multiple times throughout the program.
   Eg. Def test(a,b):
   Return a+b
2) Function Call: A function call is an instance of executing a function. When a function is called, The program execution jumps to the start of the function, runs the code inside the body of the function and then returns to the location in the code where the function was called. During a function call, you can pass parameters to the function and receive a return value from the function.
   Eg. Test(1,2)

5. **How many global scopes are there in a Python program? How many local scopes?**
   Answer: In a Python program, there is only one global scope. The number of local scopes can vary, depending on the number of functions and blocks of code you have in your program.
   A global scope refers to the names of variables which are defined in the main body of a program. These are visible and accessible throughout the program. Whereas Local scope refers to the names which are defined within a function and are local to that function.

6. **What happens to variables in a local scope when the function call returns?**
   Answer: When a function call returns, the variables defined within that local scope are destroyed. Once a local scope is destroyed any variables defined within that scope become inaccessible and any data inside them is lost.

7. **What is the concept of a return value? Is it possible to have a return value in an expression?**
   Answer:
   i) A return value is the value that is returned when a function is successfully executed. It allows a function to return a result to the calling code, which can be then used for further processing or computation.
   Eg. Def add(a,b):
   Return a+b
   In the above example, the function returns the addition of the variables a and b.
   ii) Yes, It is possible to have a return value in an expression.
   Eg. Def square(x):
   return x*x
   result = square(5) + 1
   IN the above example, The function square returns the square of argument 'x'. The return value is then used as a part of a larger expression.

8. **If a function does not have a return statement, what is the return value of a call to that function?**
   Answer: If a function does not have a return statement, then Python will implicitly return a default value 'None' which a special value that represents absence of a value.

9. **How do you make a function variable refer to the global variable?**
   Answer: To make a function variable refer to a global variable we can use the 'global' keyword in the function. The global keyword allows you to access a global variable from within a function and to assign a new value to it.
   Eg.
   Def func():
   Global x
   X = "Hello"
   Func()
   Print(x+"World")

10. **What is the data type of None?**
    Answer: The datatype of 'None' in Python is 'NoneType'. The none value represents the absence of a value, and is used to indicate a function that does not return a value. Simply, It is used to define a null value or no value at all and is different from 0,false or an empty string.

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

    Answer: The sentence import areallyourpetsnamederic tries to import a module named areallyourpetsnamederic in Python. If the module exists it will be imported and contents of the module will be available for use in the program. If it doesn't then a ModuleNotFoundError will be raised.

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

    Answer: If we had a bacon() feature in a spam module, we would call it spam.bacon after importing spam. The name of the module spam acts as a namespace for the functions and variables defined within the module.

    Eg. Import spam

    Result = spam.bacon()

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

    Answer: We can make use of exception handling. We can try using try-except block to handle exceptions in code and save a programme from crashing if it encounters an error.

    Eg.

```
Try:
        Result = 1/0
Except ZeroDivisionError:
        Print("Cannot divide by zero")
```

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

    Answer:

    i) Try Clause: The purpose of the try clause is to enclose code that may raise an exception. It allows us to test a block of code for errors.

    ii) Except Clause: The except clause specifies what to do in an exception is raised within the try block. The code in except block is except block is executed when an exception is raised and provides a way to handle and recover from an exception.