**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.

1. Functions allow one to break a program into modules. This makes it easier to handle the code.
2. It implements the reuse of code. All you need to do is call the function each time you need to execute a series of statements.
3. Functions make it simple for us to adjust functionality, and multiple programmers can work on different functions.
4. It makes possible top down modular programming. In this style of programming, the high level logic of the overall problem is solved first while the details of each lower level functions is addressed later.
5. The length of the source program can be reduced by using functions at appropriate places.
6. It becomes uncomplicated to locate and separate a faulty function for further study.
7. A function may be used later by many other programs this means that a c programmer can use function written by others, instead of starting over from scratch.
8. A function can be used to keep away from rewriting the same block of codes which we are going use two or more locations in a program. This is especially useful if the code involved is long or complicated

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

**ANS:-**

The code in a function executes when the function is called, not when the function is defined.

A function is a block of code which only runs when it is called.

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

**ANS:-**

The def statement creates a function.

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

**ANS:-**

A function consists of the def statement and the code in its def clause.  
  
A function call is what moves the program execution into the function, and the function call evaluates to the function's return value.

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

**ANS:-**

There is one global scope, and a local scope is created whenever a function is called.

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

**ANS:-**

When a function returns, the local scope is destroyed, and all the variables in it are forgotten.

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

**ANS:-**

A return value is the value that a function call evaluates to. Like any value, a return value can be used as part of an expression.

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

**ANS:-**

If there is no return statement for a function, its return value is None.

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

**ANS:-**

A global statement will force a variable in a function to refer to the global variable.

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

**ANS:-**

The data type of None is NoneType.

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

**ANS:-**

That import statement imports a module named areallyourpetsnamederic. (This isn't a real Python module, by the way.)

**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:-**

Place the line of code that might cause an error in a try clause.

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

**ANS:-**

The code that could potentially cause an error goes in the try clause.  
  
The code that executes if an error happens goes in the except clause.