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

Ans :- 1. Reducing duplication of code.

2. Decomposing complex problems into simpler pieces.

3. Improving clarity of the code.

4. Reuse of code.

5. Information hiding.

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

Ans:- The code inside a function is not executed when the function is defined. The code inside a function is executed **when the function is invoked**. It is common to use the term "call a function"

1. What statement creates a function?

Ans:- def <function\_name>

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

Ans :- Use the keyword def to declare the function and follow this up with the function name. Add parameters to the function: they should be within the parentheses of the function. End your line with a colon. Add statements that the functions should execute.

Declaring function: - add(a,b):

Print(a+b)

Calling function:- add(10,20) o/p:- 30

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

Ans:- Global :- 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.

Local: - A variable is only available from inside the region it is created. This is called scope.

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

Ans:- If a function is called more than once in a program, the values stored in the function's local variables do not persist between function calls. This is because the local variables are destroyed when the function terminates, and are then re-created when the function starts again.

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

Ans:- A return is a value that a function returns to the calling script or function when it completes its task. All Python functions have a return value, either explicit or implicit.

If a return statement is used, **it must not contain an expression**

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

Ans.:- If you don't explicitly use a return value in a return statement, or if you totally omit the return statement, then Python will implicitly return a default value for you. That default return value will always be **None**

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

Ans:- To create a global variable inside a function, you can use the global keyword.

1. What is the data type of None?

Ans:- The None keyword is used to define a null value, or no value at all. None is not the same as 0, False, or an empty string. None is a data type of its own (NoneType) and only None can be None.

1. What does the sentence import areallyourpetsnamederic do?

Ans:- That import statement imports a module named areallyourpetsnamederic

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

Ans:- spam.bacon()

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

Ans:- Via Exception handling using try - except and try - except - finally

1. What is the purpose of the try clause? What is the purpose of the except clause?

Ans:- The try and except Block: Handling Exceptions. The try and except block in Python is used to catch and handle exceptions. Python executes code following the try statement as a “normal” part of the program. The code that follows the except statement is the program's response to any exceptions in the preceding try clause

try:

# Some Code

except:

# Executed if error in the

# try block