1. Def keyword​​​​​​​

In python, def defined functions are commonly used because of their simplicity. The def defined functions do not return anything if not explicitly returned whereas the lambda function does return an object. The def functions must be declared in the namespace. The def functions can perform any python task including multiple conditions, nested conditions or loops of any level, printing, importing libraries, raising Exceptions, etc.

Lambda keyword

The lambda functions can be used without any declaration in the namespace. The lambda functions are like single-line functions. These functions do not have parenthesis like the def defined functions but instead, take parameters after the lambda keyword . There is no return keyword defined explicitly because the lambda function does return an object by default.

2. The lambda keyword in Python provides a shortcut for declaring small anonymous functions. Lambda functions behave just like regular functions declared with the def keyword. They can be used whenever function objects are required.

3.The map() function:

The map() function is a higher-order function. As previously stated, this function accepts another function and a sequence of ‘iterables’ as parameters and provides output after applying the function to each iterable in the sequence.

The filter() function:

The filter() function is used to generate an output list of values that return true when the function is called.

The reduce() function:

The reduce() function applies a provided function to ‘iterables’ and returns a single value, as the name implies.

4. Function annotations are completely optional both for parameters and return value. Function annotations provide a way of associating various parts of a function with arbitrary python expressions at compile time

5. A recursive function is a function that calls itself during its execution. The process may repeat several times, outputting the result and the end of each iteration

6. Coding rules and guidelines ensure that software is:

Safe: It can be used without causing harm.

Secure: It can’t be hacked.

Reliable: It functions as it should, every time.

Testable: It can be tested at the code level.

Maintainable: It can be maintained, even as your codebase grows.

Portable: It works the same in every environment.