1. What is the relationship between def statements and lambda expressions?

🡪

|  |  |
| --- | --- |
| Def Statement | Lambda Expression |
| Can consists of any number of execution statements inside the function definition | The limited operation can be performed using lambda functions |
| To return an object from the function, return should be explicitly defined | No need of using the return statement |
| Defined using the keyword def and holds a function name in the local namespace | Defined using the keyword lambda and does not compulsorily hold a function name in the local namespace |

2. What is the benefit of lambda?

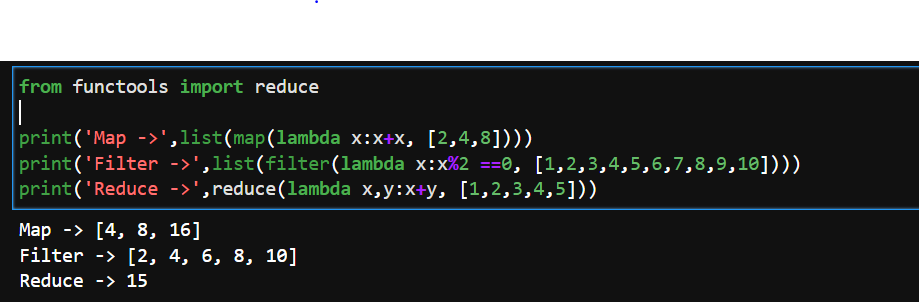
🡪 The following are some of the benefits of lambda expressions:

1. Can be used to create Nameless/Anonymous functions inside some complex functions if we are planning to use it only once.
2. Moderate to small functions can be created in a single line
3. Functions created using lambda expressions can be assigned to a variable and can be used by simply calling the variable

3. Compare and contrast map, filter, and reduce.

🡪 **map**: The map(aFunction, aSequence) function applies a passed-in function to each item in an iterable object and returns a list containing all the function call results.

**filter**: The filter function is used to create an output list consisting of values for which the function returns true  
**reduce**: The reduce function, as the name describes, applies a given function to the iterables and returns a single value



4. What are function annotations, and how are they used?

🡪 Function annotations are some random expressions which are written with the functions, and they are evaluated at compile time.

For eg: def functionName(argumentName : expression):

5. What are recursive functions, and how are they used?

🡪 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. What are some general design guidelines for coding functions?

🡪 Some of the general design guidelines for coding functions are:

* Always use a docstring to explain the functionality of the function
* Avoid using or limited use of global variables
* Proper Indentation to increase the code readability (use tab for indentation)
* Try to follow a naming convention for function names
* Avoid using digits while choosing a variable name

- Constant should be represented in all caps (ex: CONSTANT).

7. Name three or more ways that functions can communicate results to a caller.  
🡪 Some of the ways in which a function can communicate with the calling function is:

print

return

yield