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

- both are used to create functions, while def statement defines name of function and can have multiple statements, lambda functions are anonymous and one-liners. Function created by def statement can have complex logic and are reusable.

2. What is the benefit of lambda?

- lambda functions are compact hence can be used in functions like map, filter and reduce.

- we don’t have to define functions explicitly to use it hence, saves memory.

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

- map function takes the iterable and function to be applied on the iterable and returns a map object.

- filter function evaluates a condition and returns a filtered list containing elements for which the condition is True.

- reduce function reduces the whole iterable to single value. It can be used to find sum or multiplication of all the element in the iterable.

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

- function annotations are used for giving additional information about function. It tells the data type of input and output parameters. Following is example-

Def add\_numbers(a: int, b: int) -> int:

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

- the functions which calls itself in the function definition with termination condition are called as recursive functions. They are used when a complex logic is to be used repeatedly.

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

- Function Names: Choose descriptive and meaningful names for your functions that accurately reflect their purpose and functionality. Use lowercase letters with words separated by underscores (snake\_case) for consistency.

Function Length: Keep your functions concise and focused on a single task. Ideally, a function should perform one specific operation or solve one problem. If a function becomes too long or complex, consider breaking it down into smaller, reusable functions.

Function Parameters: Choose the right number and types of parameters for your function. Aim for minimal parameter lists to keep the function's interface simple. If a function requires many parameters, consider grouping related parameters into objects or dictionaries.

Function Return Values: Make sure your function returns a consistent type or a well-defined set of types. Document the expected return value and handle error cases appropriately. Avoid using return values for multiple purposes to maintain clarity.

Function Side Effects: Minimize side effects within functions. Ideally, a function should operate solely on its input parameters and produce output without modifying external states. If side effects are necessary, clearly document and handle them appropriately.

Function Documentation: Include meaningful docstrings to provide clear explanations of what the function does, its parameters, return values, and any important considerations. Follow consistent documentation standards, such as using docstring conventions like Google-style or reStructuredText.

Function Modularity and Reusability: Design functions to be modular and reusable. Encapsulate functionality into functions that can be used in multiple contexts and avoid hardcoding values or assumptions specific to a particular use case.

Function Testing: Write test cases for your functions to ensure they behave as expected in different scenarios. Use automated testing frameworks or tools to streamline the testing process.

Function Performance: Optimize function performance when necessary, but balance it with code readability and maintainability. Use appropriate data structures, algorithms, and techniques to achieve efficient code.

Consistency: Follow consistent coding style guidelines, such as adhering to PEP 8 (Python Enhancement Proposal) for Python code. Consistency in naming conventions, indentation, spacing, and overall style makes code more readable and maintainable.

7. Name three or more ways that functions can communicate results to a caller.

- return, yield statements and print, error function are the ways the function can communicate results to the caller.