**Assignment - 24**

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

Ans: The relationship between def statements and lambda expressions lies in their ability to define functions in Python. def statements are used to create named functions with a block of code, while lambda expressions are used to create anonymous functions, often for short, one-time use. Lambda expressions provide a more concise syntax for creating simple functions without needing to define a separate function using def.

1. What is the benefit of lambda?

Ans: The benefit of lambda expressions is their conciseness and simplicity. They allow for the creation of small, anonymous functions without the need for a formal function definition using def. Lambdas are particularly useful in situations where a simple function is needed as an argument to higher-order functions like map, filter, or sorted.

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

Ans: Map, filter, and reduce are all built-in higher-order functions in Python for working with iterables.

* map applies a given function to each item in an iterable and returns an iterator of the results.
* filter applies a function to each item in an iterable and returns an iterator containing only the items for which the function returns True.
* reduce (in Python 2, but moved to functools.reduce in Python 3) applies a function of two arguments cumulatively to the items of an iterable, from left to right, to reduce the iterable to a single value.

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

Ans: Function annotations are metadata about the types of arguments and the return value of functions. They are defined using colons and expressions following the parameter list and return arrow. Annotations are optional and don't affect the function's behavior but provide documentation about the function's expected inputs and outputs.

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

Ans: Recursive functions are functions that call themselves either directly or indirectly in order to solve a problem. They are used to solve problems that can be broken down into smaller, similar subproblems. Recursive functions typically consist of a base case (to terminate the recursion) and a recursive case (to call the function again with a simpler version of the problem).

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

Ans: General design guidelines for coding functions include:

* Writing functions that do one thing and do it well (single responsibility principle).
* Choosing descriptive and meaningful names for functions and parameters.
* Keeping functions short and focused (avoiding excessive nesting and complexity).
* Writing functions that are reusable and easy to understand.
* Using function annotations and docstrings for documentation.
* Following PEP 8 style guidelines for Python code.

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

Ans: Functions can communicate results to a caller through:

* Return values: Functions can return data or objects to the caller using the return statement.
* Side effects: Functions can modify mutable objects or perform actions that have an effect outside of the function's scope.
* Exception handling: Functions can raise exceptions to indicate errors or exceptional conditions to the caller.