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

**Ans:** we will use the def keyword to create a function that returns the sum of two numbers, and then we will write it as a lambda expression: As we can see, if we want our function to have multiple parameters in a lambda expression, we will just separate those parameters by a comma.

1. What is the benefit of lambda?

**Ans:** The lambda keyword 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.

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

**Ans:** Map creates a new array by transforming every element in an array, individually. filter creates a new array by removing elements that do not belong. reduce, on the other hand, takes all the elements in an array and reduces them into a single value. Just like map and filter, reduce is defined on Array.

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

**Ans:** Function annotations are a Python 3 feature that lets you add arbitrary metadata to function arguments and return value. They were part of the original Python 3.0 spec. 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.

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

**Ans:** A recursive function is a function defined in terms of itself via self-referential expressions. This means that the function will continue to call itself and repeat its behaviour until some condition is met to return a result. factorial () is a recursive function as it calls itself. When we call this function with a positive integer, it will recursively call itself by decreasing the number. Each function multiplies the number with the factorial of the number below it until it is equal to one.

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

**Ans:** Use the function naming rules lowercase with words separated by underscores as necessary to improve readability. Use one leading underscore only for non-public methods and instance variables. To avoid name clashes with subclasses, use two leading underscores to invoke Python's name mangling rules.

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

**Ans:** When the task is carried out, the function can or cannot return one or more values. There are three types of functions in Python: Built-in functions, such as help () to ask for help, min() to get the minimum value, print() to print an object to the terminal,… You can find an overview with more of these functions here.