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

Ans. Both lambda and def create the same kind of function – they have the same kind of metadata and capabilities. Their technical difference is syntactical: A lambda is an expression producing a function. A def is a statement producing a function.

1. What is the benefit of lambda?

Ans. 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.

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

Ans. Both map and reduce have as input the array and a function you define. They are in some way complementary: map cannot return one single element for an array of multiple elements, while reduce will always return the accumulator you eventually changed.

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

Ans. Function annotations provide a way of associating various parts of a function with arbitrary python expressions at compile time. The PEP-3107 makes no attempt to introduce any kind of standard semantics, even for the built-in types. All this work left to the third-party libraries.

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

Ans. A recursive function is a function in code that refers to itself for execution. Recursive functions can be simple or elaborate. They allow for more efficient code writing, for instance, in the listing or compiling of sets of numbers, strings or other variables through a single reiterated process.

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

Ans. 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

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

Ans. A function may or may not accept any argument. It may or may not return any value. Based on these facts, There are four different aspects of function calls.

function without arguments and without return value

function without arguments and with return value

function with arguments and without return value

function with arguments and with return value