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

ANS) As an expression, lambda returns a value that can optionally be assigned a name. In contrast, the def statement always assigns the new function to the name in the header, instead of returning is as a result. lambda's body is a single expression, not a block of statements.

**2. What is the benefit of lambda?**

ANS) One of the benefits of using lambda expression is the reduced amount of code.

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

ANS) **Map**

Map operation takes a mapping function and a vector of data as arguments and returns a new vector, which is the result of applying the mapping function on each element of the vector independently

**Filter**

The filter function operates on a list and returns a subset of that list after applying the filtering rule.

**Reduce**

The reduce function will transform a given list into a single value by applying a given function continuously to all the elements. It basically keeps operating on pairs of elements until there are no more elements left.

**4. 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.They were used as a way to associate arbitrary expressions to ... For functions, you can annotate arguments and the return value.

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

ANS) 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.

When any function is called from main(), the memory is allocated to it on the stack. A recursive function calls itself, the memory for a called function is allocated on top of memory allocated to calling function and different copy of local variables is created for each function call.

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

ANS)

* Limited use of globals
* Standard headers for different modules
* Naming conventions for local variables, global variables, constants and functions
* Indentation
* Error return values and exception handling conventions:

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

ANS)

* def fun2():
* print ("Called by fun1()")
* def fun1(): # function definition.
* print ("Called by main function")
* fun2() # calling fun2() from fun1()
* fun1() # calling a function.