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

**Lambda expression contains a single expression while a normal function with def statements may contain multiple expressions. Lambda expression provides an object which can later be used to return the value of the expression it contains.**

**While we define a function using the keyword ‘def’ followed by the name of the function, there is no such name in case of lambda**

2. What is the benefit of lambda?

**One of the most benefits of a lambda expression is to reduce the amount of code. It is faster**

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

**The map() function iterates through all items in the given iterable and executes the function on each of them.**

**Similar to map(), filter() takes a function object and an iterable and creates a new list. However, the new list that contains only elements that satisfy a certain condition**

**reduce() works differently than map() and filter(). It does not return a new list based on the function and iterable we've passed. Instead, it returns a single value**

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

**One of the issues with dynamically typed language is that type errors are caught only in the run-time. Python provides a way to handle this with the help of annotations.**

**The argument name is followed by ‘:’ which is then followed by the expression. Annotation syntax is shown below.**

**def foobar(a: expression, b: expression = 5):**

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

**A recursive function is a function that calls itself during its execution. Every recursive function must have at least two cases. They are :**

**The Base Case which leads to the recursion to end**

**The Recursive Case is the more general case of the problem we are trying to solve, using a recursive call to the same function.**

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

**# A function should be small because it is easier to know what the function does**

**# A function should complete only one task, not multiple tasks**

**# contain code with the same level of abstraction**

**# have fewer than 4 arguments**

**# have no duplication**

**# use descriptive names**

**# Use docstrings**

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

**a) simply type the function name with appropriate parameters**