**1. What is the relationship between def statements and lambda expressions ?**  
ANS - def statements are used to make the normal functions whereas lambda expresion is used to make the Anonymus function. which can be assigned to a variable and can be called using the variable later in function.  
Lambda is a single line functions and not the blocks of the statements. The lambda expression's body is similar to what we'd put in a def body's return statement. We simply type the result as an expression instead of explicitly returning it. Because it is limited to an expression, a lambda is less general than a def statement.

**2. What is the benefit of lambda?**  
ANS - Can be used to create an Annonymus Function.  
It require fewer line of coding.  
Fuctions created using lambda expressions can be assigned to a variable and can be used by simply calling the variable.  
Small functions can be created in a single line.

**3. Compare and contrast map, filter, and reduce.**  
ANS – **map()**: The map() function is a type of higher-order. This function takes another function as a parameter along with a sequence of iterables and returns an output after applying the function to each iterable present in the sequence.  
**filter():** The filter() function is used to create an output list consisting of values for which the function returns true.  
**reduce():** The reduce() function, as the name describes, applies a given function to the iterables and returns a single value

**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 pythoncexpressions at compile time.  
Annotations of simple parameters def func(x: expression, y: expression = 20):  
Whereas the annotations for excess parameters are as − def func (\*args: expression, \*kwargs: expression)

**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.

**6. What are some general design guidelines for coding functions?**  
ANS - 1. Avoid using digits while choosing a variable name 2. avoid using or limited use of global variables  
3. Proper Indentation to increase the code readability  
4. try to use a name for the function which conveys the purpose of the function  
5. Always use a docstring to explain the functionality of the function 6. try to follow a naming convention for function names (pascal Case or camelCase) and stick with the same convention throughout the application  
7. Local variables should be named using camelCase format (ex: localVariable) whereas Global variables names should be using Pascal Case (ex:GlobalVariable).

**7. Name three or more ways that functions can communicate results to a caller.**  
ANS - Some of the ways in which a function can communicate with the calling function is:  
**print  
return  
yield**