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

**Ans.** def statement is used to create a normal function. Where as lamba expressions are used to create anonymous functions. Which can be assigned to a variable and can be called using the variable later in function.

Lambda's body is a single expression and not a block of statements like def statement.

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

**Ans.** The following are some of the benefits of lambda expressions:

\* Can be used to create Nameless/Anonymous functions inside some complex functions if we are planning to use it only once.

\* Moderate to small functions can be created in a single line.

\* Fuctions created using lambda expressions can be assigned to a variable and can be used by simply calling the variable.

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

**Ans.**The difference between map, filter and reduce are:

\*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 are arbitrary python expressions that are associated with various part of functions. These expressions are evaluated at compile time and have no life in python's runtime environment. Python does not attach any meaning to these annotations.

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, showing the result and the end of each iteration.

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

**Ans.** Some of the general design guidelines for coding functions are:

\* Always use a doc string to explain the functionality of the function.

\* Avoid using or limited use of global variables.

\* Proper Identation to increase the code readability.

\*Try to follow a naming convention for function names (pascalCase or camelCase) and stick with the same convention throughout the application.

\* Avoid using digits while choosing a variable name.

\* Try to use a name for the function which conveys the purpose of the function.

\*Local variables should be named using camelCase format (ex: localVariable) whereas Global variables names should be using PascalCase (ex:GlobalVariable).

\* Constant should be represented in allcaps (ex:CONSTANT).

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

**Ans.**The ways through which functions communicate results to a caller are:

\*return

\* print

\* yield