Assignment 24

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

**Lambda functions:** are operator can have any number of arguments, but it can have only one expression. It cannot contain any statements and it returns a function object which can be assigned to any variable. They can be used in the block they were created.

**def functions:** Functions help break our program into smaller and modular chunks. As our program grows larger and larger, functions make it more organised and manageable. They can be called and used anywhere we want.

2. What is the benefit of lambda?

An anonymous function is a [function](https://www.programiz.com/python-programming/function) that is defined without a name.

While normal functions are defined using the def keyword in Python, anonymous functions are defined using the lambda keyword.

Hence, anonymous functions are also called lambda functions.

We use lambda functions when we require a nameless function for a short period of time.

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

Lambda functions are used along with built-in functions like filter(), map() and reduce().

The **filter() function** in Python takes in a function and a list as arguments. The function is called with all the items in the list and a new list is returned which contains items for which the function evaluates to True.

The **map() function** in Python takes in a function and a list. The function is called with all the items in the list and a new list is returned which contains items returned by that function for each item.

The **reduce() function** facilitates a functional approach to Python programming. It performs

a rolling-computation as specified by the passed function to the neighboring elements, by taking a function and an iterable as arguments, and returns the final computed value.

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

* Function annotation is the standard way to access the metadata with the arguments and the return value of the function.
* These are nothing but some random and **optional** Python expressions that get allied to different parts of the function.
* They get evaluated only during the compile-time and have no significance during the run-time of the code.
* They do not have any significance or meaning associated with them until accessed by some third-party libraries.
* They are used to type check the functions by declaring the type of the parameters and the return value for the functions.
* The string-based annotations help us to improve the help messages.

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

A recursive function is a [function](https://techterms.com/definition/function) that calls itself during its execution. The process may repeat several times, outputting the result and the end of each [iteration](https://techterms.com/definition/iteration).

Recursive functions allow programmers to write efficient [programs](https://techterms.com/definition/program) using a minimal amount of code. The downside is that they can cause infinite loops and other unexpected results if not written properly.

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

* Any args or input parameters should be placed within these parentheses
* The function first statement can be an optional statement- docstring or the documentation string of the function
* The code within every function starts with a colon (:) and should be indented (space)
* The statement return (expression) exits a function, optionally passing back a value to the caller. A return statement with no args is the same as return None.

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

Some of the ways that function can communicate results to a caller are:

1. print
2. return
3. yield