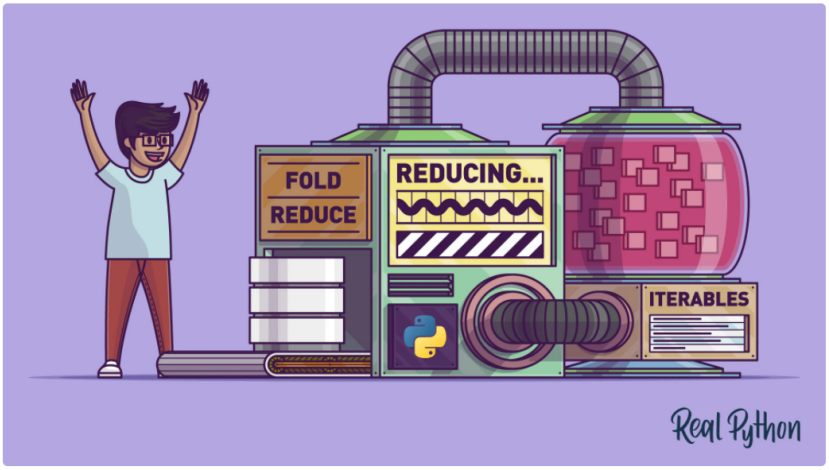

Python's reduce(): From Functional to Pythonic Style

by Leodanis Pozo Ramos Jun 29, 2020 5 Comments best-practices intermediate python

Mark as Completed

Table of Contents

Python's `reduce()` is a function that implements a mathematical technique called **folding** or **reduction**. `reduce()` is useful when you need to apply a function to an iterable and reduce it to a single cumulative value. Python's `reduce()` is popular among developers with a **functional programming** background, but Python has more to offer.

In this tutorial, you'll cover how `reduce()` works and how to use it effectively. You'll also cover some alternative Python tools that can be more Pythonic, readable, and efficient than `reduce()`.

In this tutorial, you'll learn:

- How Python's `reduce()` works
- What the more common reduction **use cases** are

Table of Contents
- Exploring Functional Programming in Python
- Getting Started With Python's reduce()
- Reducing Iterables With Python's reduce()
- Comparing reduce() and accumulate()
- Considering Performance and Readability
- Conclusion

Mark as Completed

Tweet Share Email

- How to **solve** these use cases using `reduce()`
- What **alternative Python tools** are available to solve these same use cases

With this knowledge, you'll be able to decide which tools to use when it comes to solving reduction or folding problems in Python.

For a better understanding of Python's `reduce()`, it would be helpful to have some previous knowledge of how to work with Python iterables, especially how to loop over them using a `for` loop.

Exploring Functional Programming in Python

Functional programming is a programming paradigm based on breaking down a problem into a set of individual functions. Ideally, every function only takes a set of input arguments and produces an output.

In functional programming, functions don't have any internal state that affects the output that they produce for a given input. This means that anytime you call a function with the same set of input arguments, you'll get the same result or output.

In a functional program, input data flows through a set of functions. Each function operates on its input and produces some output. Functional programming tries to avoid mutable data types and state changes as much as possible. It works with the data that flow between functions.

Other core features of functional programming include the following:

- The use of **recursion** rather than loops or other structures as a primary flow control structure
- A focus on lists or arrays processing
- A focus on *what* is to be computed rather than on *how* to compute it
- The use of pure functions that avoid **side effects**
- The use of higher-order functions

There are several important concepts in this list. Here's a closer look to some of them:

- **Recursion** is a technique in which functions call themselves, either directly or indirectly, in order to loop. It allows a program to loop over data structures that have unknown or unpredictable lengths.

- **Pure functions** are functions that have no side effects at all. In other words, they're functions that do not update or modify any global variable, object, or data structure in the program. These functions produce an output that depends only on the input, which is closer to the concept of a mathematical function.

- **Higher-order functions** are functions that operate on other functions by taking functions as arguments, returning functions, or both, as with Python decorators.

Since Python is a multi-paradigm programming language, it provides some tools that support a functional programming style:

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed

above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van

Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

>

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as [first-class objects](#)
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects

- [Recursion](#) capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects

- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities

- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities

- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`

- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities

- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`

- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`

- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators

- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators

- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`

- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators

- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`

- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python [isn't heavily influenced](#) by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in

1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed above.

In response, several functional tools were added to the language. According to Guido van Rossum, they were contributed by a community member:

> Python acquired `lambda`, `reduce()`, `filter()` and `map()`, courtesy of (I believe) a Lisp

- Functions as first-class objects
- Recursion capabilities
- Anonymous functions with `lambda`
- Iterators and generators
- Standard modules like `functools` and `itertools`
- Tools like `map()`, `filter()`, `reduce()`, `sum()`, `len()`, `any()`, `all()`, `min()`, `max()`, and so on

Even though Python isn't heavily influenced by functional programming languages, back in 1993 there was a clear demand for some of the functional programming features listed